AFRL-SN-WP-TR-2006-1039

HELPR: HYBRID EVOLUTIONARY LEARNING FOR PATTERN RECOGNITION

Mateen M. Rizki and Louis A. Tamburino

Wright State University
Department of Computer Science and Engineering
Dayton, OH 45435

DECEMBER 2005

Final Report for 30 April 1999 – 30 November 2005

Approved for public release; distribution is unlimited.

STINFO FINAL REPORT

SENSORS DIRECTORATE
AIR FORCE RESEARCH LABORATORY
AIR FORCE MATERIEL COMMAND
WRIGHT-PATTERSON AIR FORCE BASE, OH 45433-7320

NOTICE

Using Government drawings, specifications, or other data included in this document for any purpose other than Government procurement does not in any way obligate the U.S. Government. The fact that the Government formulated or supplied the drawings, specifications, or other data does not license the holder or any other person or corporation; or convey any rights or permission to manufacture, use, or sell any patented invention that may relate to them.

This report was cleared for public release by the Air Force Research Laboratory Wright Site (AFRL/WS) Public Affairs Office (PAO) and is releasable to the National Technical Information Service (NTIS). It will be available to the general public, including foreign nationals.

PAO Case Number: AFRL/WS-06-0419, 15 Feb 2006.

THIS TECHNICAL REPORT IS APPROVED FOR PUBLICATION.

/S/

Ken Grier
Project Engineer
ATR & Fusion Algorithms Branch
Sensor ATR Technology Division

/S/

Dale E. Nelson, Ph.D.
Chief, ATR & Fusion Algorithms Branch
Sensor ATR Technology Division
Sensors Directorate

/S/

Patrick D. Kee
Lt. Col., USAF
Deputy, Sensor ATR Technology Division
Sensors Directorate

This report is published in the interest of scientific and technical information exchange and its publication does not constitute the Government’s approval or disapproval of its ideas or findings.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
December 2005	Final	04/30/1999 – 11/30/2005

4. TITLE AND SUBTITLE
HELPR: HYBRID EVOLUTIONARY LEARNING FOR PATTERN RECOGNITION

5a. CONTRACT NUMBER
F33615-99-C-1441

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER
62204F

6. AUTHOR(S)
Mateen M. Rizki and Louis A. Tamburino

5d. PROJECT NUMBER
6095

5e. TASK NUMBER
04

5f. WORK UNIT NUMBER
03

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Wright State University
Department of Computer Science and Engineering
Dayton, OH 45435

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Sensors Directorate
Air Force Research Laboratory
Air Force Materiel Command
Wright-Patterson AFB, OH 45433-7320

10. SPONSORING/MONITORING AGENCY ACRONYM(S)
AFRL/SNAT

11. SPONSORING/MONITORING AGENCY REPORT NUMBER(S)
AFRL-SN-WP-TR-2006-1039

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES
Report contains color.

14. ABSTRACT

The availability of inexpensive sensors coupled with the rise of the internet has led to a rapid expansion in the amount of data available for analysis. Although there are a myriad of uses for this data, one of the most common applications is pattern recognition. The traditional approach to creating pattern recognition systems is human intensive requiring experts with training in pattern recognition to collaborate with experts who have knowledge of the problem domain to develop a custom recognition system for a specific problem. In contrast, the HELPR software architecture has focused on the design and implementation of software tools and techniques that use evolutionary computation to synthesize target systems from raw data, thereby reducing the personnel requirements and time needed to deploy new recognition systems. Result produce by ATR systems evolved using HELPR are reported for a variety of tasks involving HRR, SAR, E3D, and image processing.

15. SUBJECT TERMS
Neural Networks, Evolutionary Algorithms, Pattern Recognition, Feature Extraction, Feature Selection, Optimization

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT:	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON (Monitor)
a. REPORT	b. ABSTRACT	c. THIS PAGE			Kenneth A. Grier
Unclassified	Unclassified	Unclassified	SAR	116	19b. TELEPHONE NUMBER *(Include Area Code)*

Contents

List of Figures

List of Figures

List of Tables

Preface

Many people have contributed their time and talent to the development of HELPR. Most significantly, Dr. Michael Zmuda was involved in almost every aspect of the design and implementation of HELPR including all the work related to HRR, SAR, and E3D data analsysis. His expertise in mathematical morphology and software engineering was critical to the success of the HELPR project. In addition, Dr. Dale Courte made major contributions to the work related to signal analysis and later collaborated on the color constancy problem. The contact information for these individuals is given below.

Professor Michael Zmuda
Miami University
Department of Computer Science and System Science
230K Kreger Hall
Oxford, Ohio 45056

Professor Dale Courte
University of Dayton
Department of Computer Science
147 Anderson Hall
Dayton, Ohio 45469

1.0 Summary

This research effort defined, developed, and implemented a test-bed for the automatic synthesis of target recognition systems. The approach orchestrates a hybrid evolutionary learning process to grow feature detectors, assemble sets of cooperative features, and construct an accurate classification system. Key aspects of this approach include: (1) a multifaceted representation that defines a rich space of potential recognition system designs, (2) a closed-loop learning system that synthesizes recognition systems by integrating multiple evolutionary learning algorithms, (3) the use of constructive synthesis to evolve complex processing networks from simple networks as needed, and (4) a modular implementation that facilitates the use of alternative approaches for different phases of the target recognition problem. This effort leverages significant research and development expertise in the areas of pattern recognition, evolutionary computation, morphological analysis, program management, and software engineering for the design and implementation of defense applications. The technology and methods produced as a result of this program promote the rapid construction of recognition systems for Air Force applications; thereby reducing the cost, manpower requirements, and time needed to deploy new systems. The operation of the test-bed are validated using HRR, SAR, and E3D data sets.

2.0 Introduction

2.1 Background/Scope

A pattern recognition system (PRS) that solves a complex problem is an inherently complex structure consisting of multiple, interacting components such as features, a feature set and a classifier [15, 16]. In most applications, primitive features are given and the task of forming a recognition system reduces to assembling feature sets and a classifier. A more challenging problem involves synthesizing primitive features and then assembling the final stages of the recognition system. The difficulty in synthesizing complete recognition system is due in part to a trade-off that exists between features and classifiers. Good features reduce the need for a complex classifier, while a good classifier can compensate for marginal features. Researchers have developed

expert systems and pure optimization techniques to semi-automate the design of PRSs. Although reducing the time needed to develop a recognition system, these techniques still require significant time, cost, and human expertise to develop solutions for difficult ATR problems involving SAR and HRR data.

2.1.1 Expert Systems

Expert systems have been developed to assist in the development of algorithms for signal or image classification. Vogt developed the REM system that synthesizes size-limited image processing programs to map input images to acceptable output images [64]. REM encoded rules to exploit the algebraic properties of the available operators. However, REM's rule set was unable to create programs consisting of more than two operators due to a lack of intermediate performance measures to guide the assembly of operators. Researchers have developed expert systems that generate larger programs by requiring an analyst to precisely specify the objective using natural language [49], predicate logic [33], abstract programming language [37], or grammars [13]. Unfortunately, defining a specification is a very difficult task similar to traditional software development.

The HELPR approach does not use a rule base for constructing recognition programs. Instead, the technique uses evolutionary search techniques to assemble operators into a complex system. Information theoretic performance measures guide the assembly process toward successful configurations. By using generalized performance measures such as entropy, Fischer's discriminant [20], confidence values, and orthogonality the HELPR can find high-quality solutions that might otherwise be eliminated by human biases.

2.1.2 Pure Optimization

Optimization has been thoroughly studied for many decades and includes gradient-based [10]; stochastic [1, 29, 50]; neural networks [3]; fuzzy sets [69]; and population-based techniques [1, 29, 50]. Most pure optimization algorithms apply only to problems of identifying unknown scalars or vectors of scalars, and are not directly applicable to the problem of constructing a complex computational structure. In order to apply an optimization technique to the problem of designing a recognition system, researchers must specify the overall structure of the system and

then apply the algorithm to identify parameter values that maximize criteria such as recognition rates or efficiency. Creating a system using pure optimization has several deficiencies. First, determining the general structure of the system requires significant human expertise. Second, structures defined by humans are confined to a small subset of the entire search space and therefore may bias the search toward potentially sub-optimal solutions. Lastly, rapid development of recognition algorithms is impractical due to slow turnaround time.

Researchers have applied pure optimization to the design of PRSs, where components, structures, and/or functional forms are specified *a priori*. Neural networks are widely used to classify fixed-size feature sets. These technique start with a fixed topology and simple adjust synaptic weights to optimize an objective function. Evolutionary techniques have been used to optimize specific compounds of recognition systems including convolution kernels [34]; binary structuring elements [27, 54, 55, 68]; grayscale structuring elements [68]; and feature sets [40, 52, 57].

2.1.3 Synthesis

Creating a complex computational structure such as a recognition system requires synthesizing a structure to produce a desired response. Whereas pure optimization identifies parameters that optimize the behavior of a pre-specified structure, synthesis techniques must create the optimal configuration of operators, in addition to identifying the configuration's optimal parameter values. Consider the issues associated with forming a canonical pattern recognition system composed of a feature extraction module followed by a classification module. How many detectors should be included in the detector set? What types of operations should be used in each feature detector? What is the correct number of operations to include in each detector? Which operator sequences form useful functions? These questions involve structure-function relationships that can only be addressed by synthesizing each component of a recognition system. HELPR's synthesis process has the following properties:

1. The system's computational structure is initially unspecified.
2. The resulting system attains the appropriate level of complexity. Simple recognition tasks result in small systems; whereas, complex tasks will result in larger, more complex solutions.
3. A computationally minimal set of features is formed.

4. The proper sequence of operators to include in each feature detector is automatically determined.
5. Operator sequences that perform generic functions are stored in a reusable form.

2.1.4 Synthesis Techniques

Synthesis techniques were first investigated during the mid 1960's when researchers used evolutionary computation to generate finite state automata [22, 62]. More recently, specialized architectures such as the Neocognitron [23] and Cascade Correlation (CC) [17] have been developed to overcome the limitations of fixed-architecture neural networks [3]. Although these networks grow in size, the network topology is still restricted.

Genetic Programming (GP) [36] represents an unknown computational structure as an expression tree with each node representing an operator. The tree representation is extremely flexible and therefore is not limited to configurations defined by experts. GP has successfully synthesized structures for complex problems, but its direct application to the design of a recognition system is limited for two reasons. First, GP is designed to synthesize one monolithic structure. This makes it difficult to use this approach to synthesize a cooperative collection of distinct feature detectors. Second, GP does not directly identify useful building blocks or sequences of operators. Our approach uses a modified GP algorithm in concert with other evolutionary learning algorithms to grow a multiplicity of feature detectors while explicitly measuring the usefulness of building blocks. This accelerates the search process by eliminating the computational costs associated with evaluating less useful structures.

A multi-faceted evolutionary system, E-MORPH [45, 46], has been used to generate pattern recognition systems for classification tasks such as HRR [47]. E-MORPH uses GAs to optimize feature sets; EP to synthesize structuring elements; and GP to optimize topology. The approach generalizes and extends the techniques developed in the E-MORPH system.

2.1.5 Evolutionary Computation for PRS Synthesis

Algorithms from the field of evolutionary computation (EC) have been used in synthesis. The field of EC consists of four main approaches: Genetic Algorithms (GA) [29], Evolutionary

Programming (EP) [21], Genetic Programming (GP) [36], and Evolutionary Strategies (ES) [50]. All four EC techniques share general characteristics:

1. Stochastic search technique
2. Population-based search technique
3. Exhibit good performance in high-dimensional multi-modal search spaces

The appropriateness of each of these paradigms depends on the intended application. For example, the GA excels when the problem can be cast as the optimization of a fixed number of scalars (i.e., pure optimization). Additionally, GAs require that the search space be represented using a natural genetic encoding such as a bit string. GP is similar to the GA but represents the solution space as arbitrarily sized trees and therefore is appropriate for tasks requiring the synthesis of structures that can be represented as expression trees. EP, like GA, is appropriate for optimization tasks. In contrast to the GA, EP places no emphasis on the underlying genetic representation, and therefore it is a good technique for optimizing real-values or when a natural genetic encoding is unknown. ES distinguishes itself from other EC techniques in that the genetic representation includes information for controlling the search technique itself. Thus, ES includes meta information directly into the search space and reduces the number of system parameters (e.g., mutation rate).

3.0 Methods, Assumptions, and Procedures

The process of pattern recognition is usually conducted in several distinct stages starting with transformation of raw input data, identification and extraction of features, formation of features sets, and classification. Each stage of this pipeline transforms and/or reduces the flow of information to facilitate the actions performed in the subsequent stages. The initial transformations eliminate noise and reorganize the data flow to simplify the identification of features. The process of feature identification and extraction produces a pool of measurements that ideally are class invariant. Feature selection is the process of finding a small subset of features sufficient to discriminate among different classes of data samples. Lastly, classification associates a user

defined label with the various combinations of feature-values. Overall, the process of forming a recognition system consists of reducing high-dimensional input data to simple categorical values.

Many researchers have attempted a "black box" approach to the problem of synthesizing complex recognition systems. These techniques are strictly tied to terminal performance measures of recognition accuracy, utilize a homogeneous representation of the underlying system, and are unsuccessful as the complexity of the recognition task increases. In contrast, we view a pattern recognition system as a series of modular components that are designed to perform distinct tasks. Consequently the representation and approach used to synthesize each component is customized to the modules' objectives, yet the flow of information between components is carefully orchestrated to achieve the overall goal of producing an accurate recognition system.

HELPR addresses the problem of synthesizing and integrating the individual components of a recognition system. Our approach uses a multifaceted representation to define a search space consisting of layers of computational networks that are synthesized using a multiplicity of evolutionary learning paradigms. An overview of the approach is shown in Figure 1. A recognition system consists of four basic modules: transformation, feature identification and extraction, feature selection, and classification. The identification and extraction module is further divided into a primitive and complex feature extraction components. Transformations use a network of morphological operators, called a Morphonet, that is synthesized using a modified form of GP. The input to this component is raw signals or images and the output is a collection of transformed data flows. The first phase of feature extraction, labeled reduction, filters the transformed data to extract statistical and geometric relationships among subsets of data flows. This process samples or taps the various data flows to reduce the high-dimensional information to a scalar. The reduction operators are synthesized using a modified version of EP. In the second phase of feature extraction, a collection of increasing complex, nonlinear features are formed using another GP process. Feature selection is then accomplished using a GA. In the final stage, feature sets are attached to neural network classifiers for evaluation. Within this framework, the many subsidiary evolutionary processes work to assemble and optimize each type of component. The entire process is driven by a top-level resource allocation and control strategy that uses information pertain-

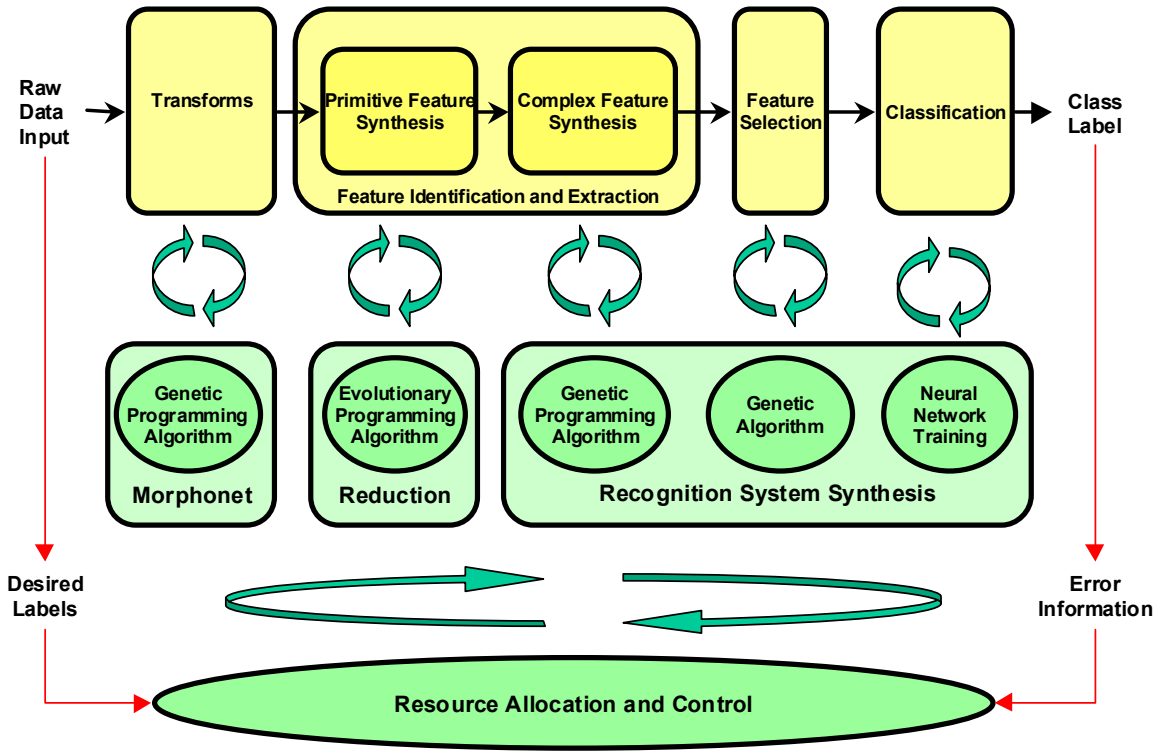


Figure 1. Overview of the HELPR Approach to PRS Synthesis.

ing to the user's desired classification and error information to drive the formation of improved sub-components.

3.1 Keys to Effective Use of Evolutionary Learning

Evolutionary learning techniques are stochastic processes. A collection of partial designs defines a population of design alternatives. Each modification to an existing design represents another alternative. Application of an evolutionary learning process simply produces generations of alternative designs that are modifications, usually improved, of earlier forms. Designs are evaluated relative to prescribed goals and constraints and those not satisfying the design requirements are eliminated. There are four basic issues that must be addressed to evolve and integrate the various components of a pattern recognition system. These issues include:

1. defining a representation for each component design
2. choosing search techniques to explore the alternatives defined by each representation

3. constructing performance measures (goals and constraints) to guide the synthesis of individual components
4. incorporating feedback and control mechanisms to orchestrate the search processes among the different components to produce a complete system

These items guide the following discussion of the design and implementation of individual components that form the basis for the hybrid evolutionary learning system.

3.2 Morphonet

Morphological analysis is used to transform the raw input data into a form suitable for primitive feature extraction. The representation is referred to as a Morphonet because the operations are based on the principles of mathematical morphology [30, 51]. Mathematical morphology is a technique for probing the structure of signals or images using set theoretic operations. Each morphological operation is a signal-to-signal or image-to-image transformation that applies a probe-like pattern, called a structuring element, to an input data flow, to isolate or enhance characteristics of the data.

The task of formulating morphological expressions is difficult even for an experienced morphological analyst, but if the correct algebraic form and structuring elements are brought together it is possible to solve complex recognition tasks. Observe the effect of applying a handcrafted morphological expression to three signals shown in Figure 2.

In this example, a small set of signals is processed using a single morphological transform (BandClose Ball 3 5) which is equivalent to (- (Close Ball5) (Close Ball3)). Each signal is first closed with a ball of radius 5. A closing operation removes peaks with a base size smaller than the radius of the ball. By taking the difference between two closing operations, only certain size peaks are allowed to pass through. After applying the band operator, a trivial discriminant function could easily separate the three signals. Handling more extensive variations or a larger variety of signals requires a more elaborate morphological expression. Detailed examples of mor-

phological operations, structuring elements, and the process of generating expressions are described in [70, 71].

Early work with morphological analysis demonstrated its value as the basic algebra for defining expressions for pattern recognition [40, 41, 58, 59, 60]. In the MORPH [72] system we used evolutionary computation to synthesize morphological target detectors for optical imagery. This technique was expanded in E-MORPH [45, 46] to form parallel flows of morphological features for multi-class pattern recognition.

In [47] we demonstrated that GP is a viable approach to synthesizing morphological transformations for pattern recognition. This system used the traditional GP representation of expression trees to form the morphological structures. There are two problems, however, associated with using the standard GP tree representation. First, useful pieces of transformations (sub-expressions) are duplicated throughout the population of recognition systems. Evaluating these superfluous expressions is computationally expensive; limits the system's ability to explore new structures; and fails to localize useful structures for reuse. Another limitation stems from GP's tendency to grow structure from the top -- down. GP exchanges sub-expressions at random locations in pairs of trees to form new expressions. Consequently, there is a bias toward exchanging the most deeply nested functions in the expression. This type of search is ineffective when the low-level functions tend to reorganize and reduce information flow as is typical in the initial stage of a target recognition system. After an expression is evaluated and accepted into the system, random changes in their support functions may be useful in arithmetic problems but tend to be disruptive in image and signal processing applications.

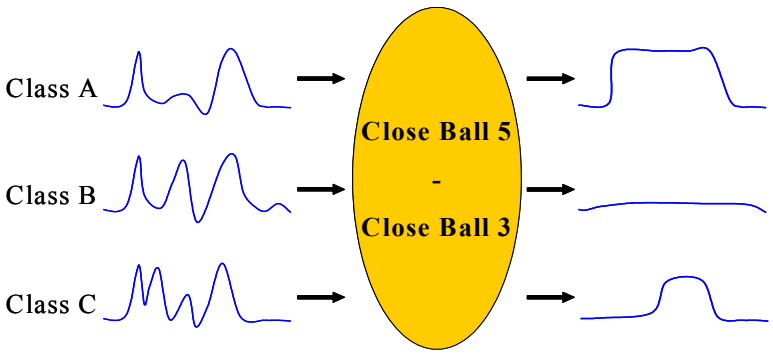


Figure 2. Sample Morphological Operator.

The Morphonet structure is a directed acyclic graph composed of morphological processing elements. The form of the Morphonet is shown in Figure 3. Input data flows through functional nodes (numbered 1-19) to produce a collection of transformed outputs. Each connected set of nodes defines an expression. For example, the expression terminating at node number 16 is $F_{16}(F_{12}(F_5(\text{Input})),F_6(\text{Input}))$. Notice the Morphonet representation maintains only one copy of each sub-expression, eliminating the disadvantages of maintaining redundant structures.

Wavelets are an alternative technique for reorganizing the initial data flow. We have used Gabor wavelets [26] to transform optical images in a multi-resolution stack of grayscale images [43]. In [44] we developed a static network of morphological expressions to perform a similar multi-resolution decomposition of the input. These techniques require the user to select the type of wavelet or structuring elements, number of resolution levels to use in the decomposition and the scale of the different resolution levels. Tuning these parameters is difficult and reduces the

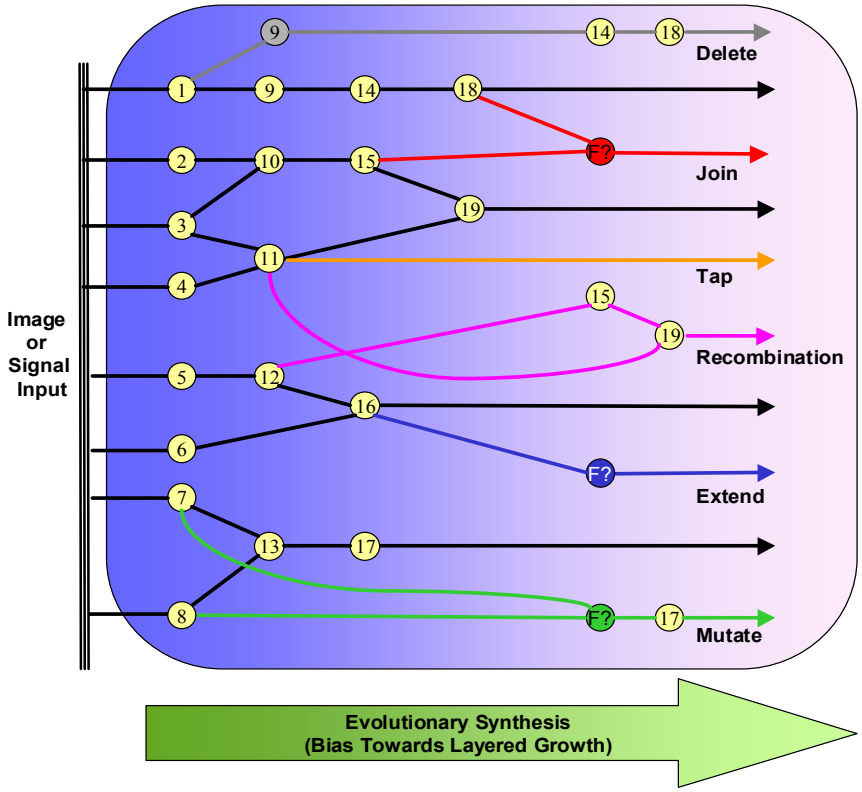


Figure 3. A Sample Morphonet.

ability to fully automate the recognition process. The parallel flows of the Morphonet are capable of defining a multi-resolution decomposition of the input.

The synthesis of the Morphonet's topology is achieved using a variation of GP. The Morphonet is a series of parallel data flows produced by applying complex morphological expressions to the input data. The transformed output of selected expressions is used as input into next stage of the recognition system. We refer to these flows as taps. The output of the remaining operations feed higher level operators within the Morphonet. The morphological transforms and their associated taps define a population that competes for survival.

To begin the evolutionary synthesis of a Morphonet, small random assemblages of morphological operators are used to transform the input data. The output of each expression is evaluated according to some intermediate performance measure to determine its fitness. The output of the fittest expressions are tapped and presented to the next stage for primitive feature extraction. An evolutionary cycle consists of reproduction with variation, evaluation, and selection. The process of reproduction creates new morphological expressions by forming modified versions of an existing expression or combining two or more pieces of existing expressions. Thus, past successful expressions become parental units producing offspring that may incorporate some of the parental structure as well as new operations. The entire search process works to synthesize the Morphonet layer-by-layer. The types of variations are shown to the far right in Figure 3.

Each expression formed during reproduction is evaluated relative to a local performance measure. One aspect of this evaluation is to prune nonviable offspring such as expressions producing constant outputs for all inputs. Unlike the evaluation techniques typically used in optimization that are geared toward maximizing or minimizing the performance of a single monolithic structure, the performance measure used to measure a Morphonet's fitness must optimize a population of expressions, while also producing a diverse collection of outputs. Our experience indicates that it is not possible to gauge the importance of an individual expression to the overall system when it is first created, since an immature individual may potentially develop into an outstanding performer despite its initially low performance. We have identified multiple criteria that must be incorporated in a local performance measure for evaluating transforms. These criteria include:

1. coverage of different regions of individual data items
2. consistency of the response within classes of data
3. variation of the response to data from different classes
4. novelty of the response relative to response of other expressions
5. computational complexity of the expression
6. age of the expression in term of cycles of evolution
7. number of primitive features that use the transform's output

Once the new expressions are evaluated, all expressions that do not feed a tap used by a primitive feature detector compete for survival.

Our previous work indicates that synthesis processes work best when growing structures from small-to-large. Consequently, the Morphonet is designed to evolve a population of transforms composed of individuals with many different levels of structural complexity. Expressions that appear early in the evolutionary process and survive tend to increase in complexity to enhance their performance. A newly formed structure may be significantly less complex and produce a marginally lower performance. The standard type of selection process used in most evolutionary algorithms would tend to underrate the importance of these simple, emerging structures. We use of a multi-tiered tournament selection algorithm that defines complexity niches [68] to overcome this problem. In a multi-tiered tournament, expressions are categorized and compete only with individuals in the same category. This is analogous to the divisions that occur in sports where high school, college, and professional teams do not interact directly, but successful individuals in each level move on to compete at more advanced levels. This approach allows emerging structures time to mature before they must compete with the full population.

In a multi-tiered tournament, individual transforms are ranked based on their performance relative to the performance of other transforms in the same category. The size of the tournament changes throughout the evolutionary process and is based on the average performance of the transforms in a given category. In these local competitions, the chance of winning is proportional to the ratio of the transform's performance to its competitor's performance. Limiting the tourna-

ments to a subset of the population reduces the possibility of premature convergence of the evolutionary process. When the average performance of the population is poor, the number of individuals in each tournament is small and a marginally better feature does not have the opportunity to dominate the population. The pair-wise competition used in tournament selection tends to maintain a diverse population by allowing marginal individuals additional time to mature. The final selection for survival is based on a ranking of the number of conflicts won by each transform. The transforms with the greatest number of victories in each category survive to the next evolutionary cycle.

3.2.1 Synthesizing primitive features

The Morphonet's output is a collection of unregistered, high-dimensional data. To identify and locate targets, primitive feature detectors must register the data flow, extract statistical and geometric relationships, and ultimately produce a scalar value.

We studied two procedures for synthesizing primitive features. The first approach is an extension of the technique developed in E-MORPH, which synthesized an array of convolution templates to register many disparate information flows in parallel. The second approach will extended the techniques developed in MORPH to synthesize small networks of special composition operators that extract features from pairs of data flows. The later approach is more sensitive to detailed information in the data flow and is as an alternative to a template based approach.

Our primary approach evolves a registered set of convolution kernels. We refer to these special convolution kernels as *caps*. There is one cap for each tapped flow in the Morphonet. A full set of caps defines a three-dimensional cap-stack that probes all of the Morphonet data flows in parallel (see Figure 4). Each cap is composed of a collection of positive and negative Gaussian shaped points that explore or tap both the geometrical structure and contrast variation of the individual transformed data flow as well as relationships between different data flows. The convolution operator produces its strongest response when all of the positive and negative probe points align with similar regions in the data flow. Consequently, the stack-cap produces a strong response when the geometry embodied in the probe also exists in the transformed data flow.

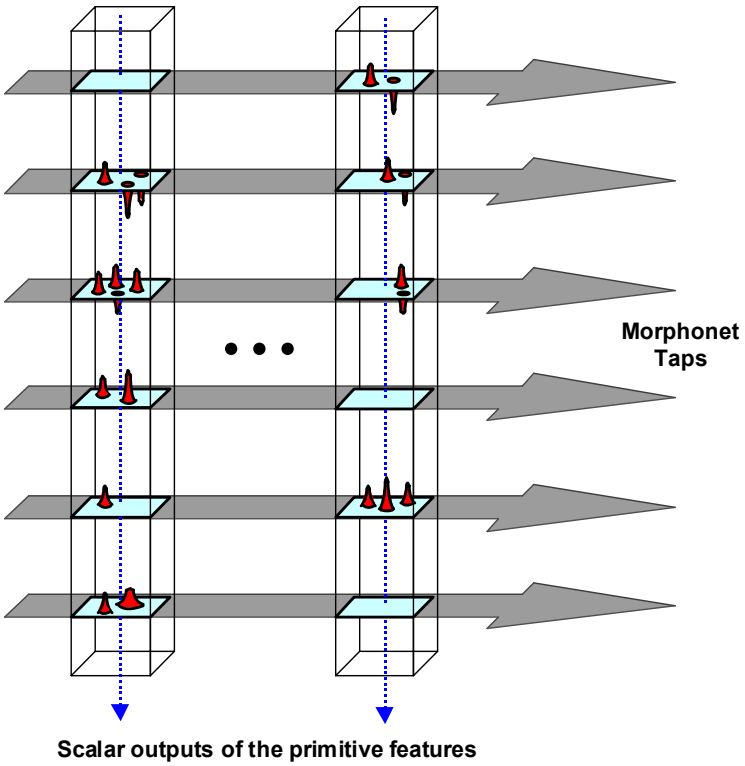


Figure 4. Representation of primitive feature detectors.

A cap-stack is represented as a set of descriptors, where each descriptor specifies the location (x,y,stack-position), variance (s), and sign of a Gaussian-shaped point. We used a modified EP algorithm to vary the structure of the caps. This technique will evolve the caps by adjusting the values and number of the descriptors. An outline of the algorithm is shown in Figure 5.

```
For each primitive feature F_i (i = 1,2,3 … N) in the population
Begin
    Clone F_i to produce an extended population F_ij (j = 1, 2, 3, … M)
    For desired number of evolutionary cycles
    Begin
        For each member of the extended population F_ij (j=1,2,3,…M)
        Begin
            Generate a mutated copy F̄_ij of F_ij
            Calculate the performance P̄_ij of F̄_ij
        End
        Apply tournament selection to rank the (M+M) values of P_ij and P̄_ij
        Select the top M ranked individuals to form the extended population
    End
    If the maximum P_ij  > original feature's (F_i) performance
        then replace F_i with F_ij
End
```

Figure 5. Algorithm for Synthesizing Features

To summarize, a primitive feature (cap-stack) is used as a seed to define an extended population. The extended population is composed of mutated copies of the parental seed that represent slightly modified design alternatives. EP is used to search for an improved version of the seed-feature. The extent of this search is restricted based on the performance of the evolving stacks. After several evolutionary cycles are complete, if the performance of the best primitive feature in the extended population exceeds the performance of the parental seed, it replaces the parent.

The types of variation applied to an individual cap are outlined in Figure 6. These genetic

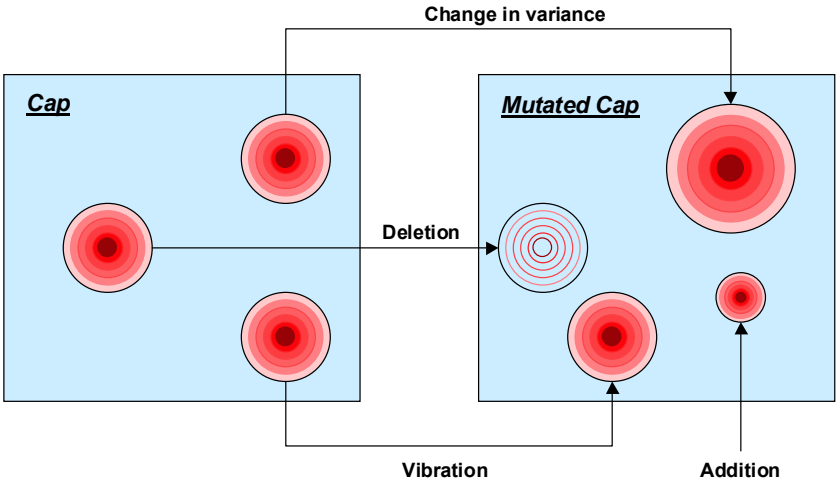


Figure 6. Technique Used to Evolve a Cap

operations adjust the position of the Gaussian points, add new points, and delete existing points. As a rule, the stack is initialized with a limited number of probe points, but the probability of addition is greater than the probability of deletion. This bias toward addition causes the stack to become increasingly sensitive. An adaptive mechanism will be used to control the amount of addition, deletion, and extent of variation of existing points. When the performance is low, the potential for variation is high. As the performance reaches a maximum level, the variation approaches zero, freezing the cap-stack's configuration. This adaptive mechanism is similar to the "cooling" process used in simulated annealing [4] that allows the search to escape local optima while driving the system toward an acceptable configuration.

We investigated several intermediate performance measures to identify those that are most effective for the task of evolving primitive feature detectors. There are a variety of measures to consider including Fisher's Discriminant [15, 16, 20], conditional class entropy [9], or mutual information [8]. These measures estimate a feature's ability to separate the data along class boundaries. The magnitude of these measures increases as the separation between the means of the response for each class of target grows while the variance in the response within each class shrinks. Feedback from later stages of the evolving recognition system can be used to guide the formation of primitive features that produce specific responses. For example, error information

produced by the population of classifiers can be used to form target response vectors. The primitive features can evolve to correlate to the desired response using performance measures such as sum squared error, root mean squared error, and confusion error.

3.2.2 Synthesizing Complete Pattern Recognition Systems

We defined a single hybrid evolutionary learning process to take the primitive features defined by the outputs of stack-caps and synthesize the final pattern recognition systems. Our approach combines GP, GAs, and neural networks to synthesize complex nonlinear features, construct cooperative sets of features, and perform classification. This technique is an extension of work described in [47].

The evolutionary process begins by forming a population of complete recognition systems as shown in Figure 7. Each recognition system contains a small random set of primitive features

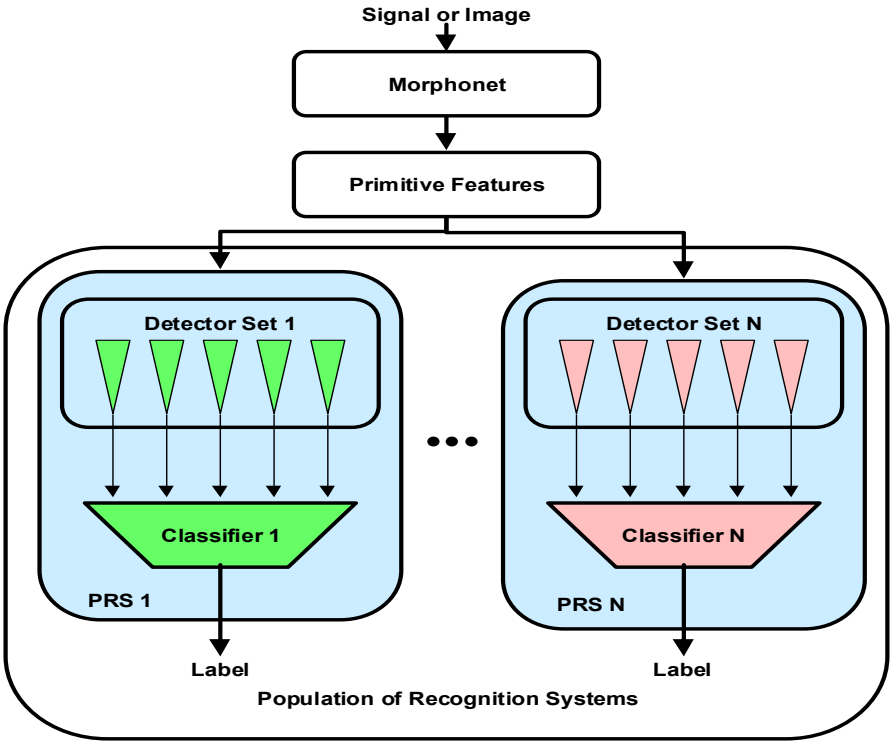


Figure 7. Representation of a Population of Pattern Recognition Systems.

sampled from the set of stack-caps. A linear Perceptron [38] is trained to evaluate the accuracy of

each recognition system. A GP algorithm is used to extend the primitive features generated by the cap-stack and a GA is used to mix detector sets. The actions of GP and GA are blended together into one evolutionary process. The GA is responsible for recombining detector sets and components of detector sets as shown in Figure 8. Parental units are selected from the base population,

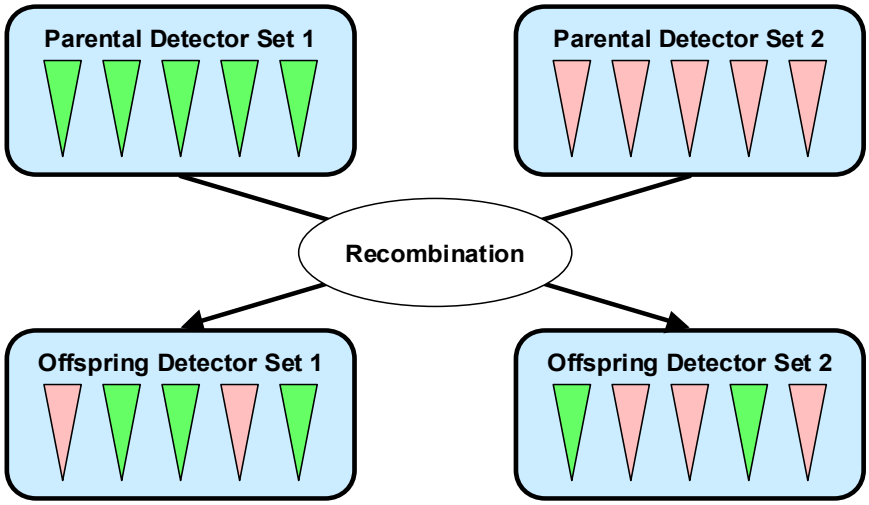


Figure 8. GA Process Used to Form Feature Sets

where the probability of selection is proportional to the recognition system's accuracy. Once a pair of parents is selected, their detectors are exchanged using a uniform crossover [5, 28]. When an exchange of detectors occurs, there is a small probability that the internal structure of the underlying expressions are also recombined using GP. The expressions are represented as trees. The input to these expressions is the output of selected cap-stacks. The GP algorithm exchanges sub-trees between pairs of complex features as shown in Figure 9. In addition to exchanging information by recombination, genetic operators exist for adding, deleting and replacing sub-trees (not shown). Mixing the structure of expressions allow the search process to explore a wide variety of new functions.

Pao [39] has shown that a linear classifier is sufficient to classify a data set if nonlinear transforms of the input are provided to the classifier. Therefore, the output of nonlinear feature detectors justifies use of simple a linear Perceptron classifier. We have previously considered the use of K-nearest neighbor classifiers for this type of problem and found that they are also effective and represent a viable alternative.

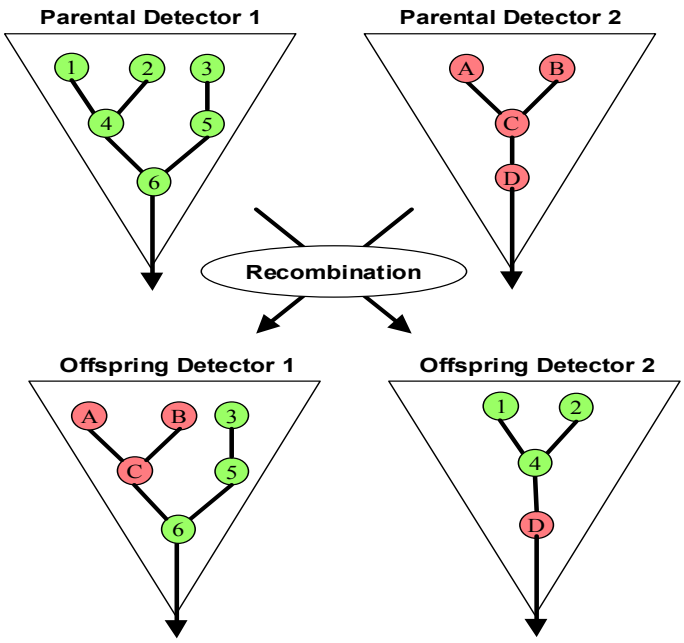


Figure 9. GP ProcessUsed to Form Complex Features.

3.2.3 Component Integration

One of the most difficult tasks associated with forming a recognition system from modular components is orchestrating all of the learning processes to form an integrated system. Two approaches to component integration will be evaluated.

First, we developed a pipeline process without direct feedback between the various evolving components. The information flows in only one direction from transforms toward the classifiers. The advantage of this approach is the components are loosely coupled making it possible to synthesize each component in series. The results of each stage of the processing can be buffered and made available to the next stage at some future time. The strength of this technique is the computational load can be readily distributed on a network of computers, virtually eliminating the problem of resource allocation. This technique is viable only if the performance measures used to generate the components are capable of predicting the specific needs of the next stage of processing. For example, in [40, 52] a large population of feature detectors was generated off-line and an

orthogonality measure applied to response vectors was used to form sets of feature detectors for classification. The downside of this technique is formulating performance measures for components located in the early stages of the pipeline that estimate contribution to terminal recognition accuracy is difficult.

Second, we developed a tightly coupled system where each component produces output for the next stage and sends requests or directives back to earlier stages. These directives are used to synthesize components that produce specific types of output. The advantage of this approach is terminal performance information such as classification accuracy indirectly guides every stage of processing. For example, the learning algorithm that evolves classifiers could post a request of the form: "generate a feature to separate class A from class B" or a more specific directive such as "generate a feature that respond to samples 1, 2, 3 in class A and does not respond to samples 2, 5, and 7 in class B". In MORPH, we developed techniques for synthesizing image transforms that used directives. A composite image was formed using the output of all transforms. Areas of low activity in the composite image were used to define regions of interest, and emerging transforms were evolved to focus on these areas. Similar techniques are incorporated into the system to guide the various phases of evolution.

Computational resources must be carefully allocated among the various learning algorithms to produce a balanced system. If too much effort is spent on any one algorithm, it may reduce the performance of the overall system. In Tamburino et al. [53, 61] we studied issues related to resource allocation applied to the problem of synthesizing recognition system components. The results of this work suggests that system performance is improved with the use of adaptive control mechanisms to partition the resource among the various algorithms. We developed a meta-level learning algorithm that monitors the performance of each component and the overall system response. The top-level controller will allocate additional computational resources to algorithms that are producing components that show no significant increase in performance, and remove resources from algorithms that produce components that improve too rapidly relative to other components. This will effectively address the complexity trade-off that occurs between features and classifiers, producing a balanced system.

3.2.4 System Validation

The system was applied to HRR, SAR and E3D data sets. Our research team worked closely with Air Force personnel to identify appropriate test data to use in experiments so results can be compared to the results obtained by other techniques.

3.3 Summary

The HELPR system uses multiple evolutionary algorithms for synthesizing pattern recognition systems. A system level diagram of the information flow is shown in Figure 10. Images or

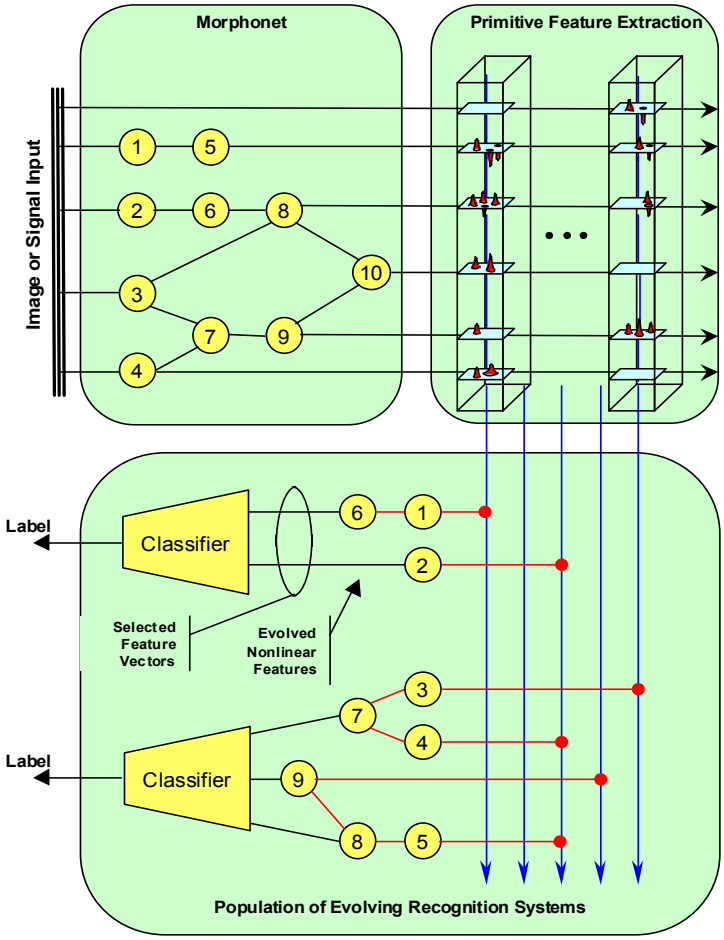


Figure 10. Information Flow in HELPR

signals flow through a Morphonet consisting of an evolved collection of morphological expressions that reorganize information to facilitate primitive feature extraction. Special 3-dimensional cap-stacks are evolved to extract primitive features. A population of recognition systems is

formed, where each recognition system is composed of a collection of complex features that are extensions of the primitive features formed in the previous stage. A combined GA/GP process is used to restructure complex features and form sets of features. Finally a linear classifier is used to label the constructed feature vectors. The evolved systems will then be used to solve recognition tasks such as HRR and SAR.

We use an extremely expressive multifaceted representation that facilitates the five distinct adaptive learning processes discussed previously are summarized in Table 1.

Table 1: Learning Processes Used in HELPR.

Adaptive Learning Process	Primary Representation	Primary Learning Algorithm	Evaluation Criteria
Synthesis of Morphonet	Directed acyclic graph of morphological operators	Genetic Programming • Delete • Join • Tap • Recombine • Extend • Mutate	• Completeness of coverage • Intra-class consistency • Inter-class variation • Novelty • Computational requirements • Age • Quantity of taps supported
Selection and optimization of primitive features	Convolution kernels	Evolutionary Programming • Variance change • Delete • Vibration	• Fischer's discriminant • Entropy
Synthesis of non-linear scalar transforms	Acyclic directed graph of arithmetic operators	Genetic Programming • Delete • Join • Tap • Recombine • Extend • Mutate	• Fischer's discriminant • Entropy
Selection of subsets of feature	Size-varying sets	Genetic Algorithm • Recombination • Set growth • Set reduction • Mutation	• Orthogonality • Computational requirements • Resulting classification errors
Formation of classifiers	Perceptron	Perceptron Training Algorithm	• Classification errors

These learning algorithms have been carefully selected based on the requirements of each task. GP is used to optimize the topology of the graphs representing computational sequences. EP and GAs are used when topology is not involved.

4.0 Results and Discussion

4.1 HRR Application

The HELPR system was used to synthesize recognition systems for HRR radar signals. XPatch was used to simulate six airborne targets across a pose window consisting of variations in azimuth of -25° to 25°and depression angle of 0 to -21 degrees. The signals were input as histograms consisting of 128 range bins. The HELPR architecture was setup to evolve morphological features to extract structural properties from the signals, a second module selected subsets of cooperative features, and the final module formed a neural network classifier. The results of these experiments demonstrated that HELPR could form recognition systems that produced recognition accuracies of 96% on independent test data using a limited number of features. The results were compared to several other techniques (perceptrons, RBF networks) that could not match HELPR's accuracy without many additional features.

To instantiate a specific version of HELPR, there are four basic issues that must be addressed. These issues include:

1. defining a representation for each component of the design
2. choosing search techniques to explore the alternatives defined by the representation
3. constructing performance measures (goals and constraints) to guide the synthesis of individual components
4. incorporating feedback and control mechanisms to orchestrate the search processes among the different components to produce a complete system

These items guide the following discussion of the design and implementation of individual components that form the basis for HELPR for HRR target recognition.

4.1.1 Representation

The representation of a HELPR generated recognition system is shown in Figure 11. Each recognition system is composed of a feature extraction module and a classification module 15, 16]. The feature extraction module applies a set of feature detectors to an input signal to form a feature vector. The set of feature detectors is viewed as a linear chromosome that defines the genotype of an individual recognition system. The classifier module then assigns a target label to

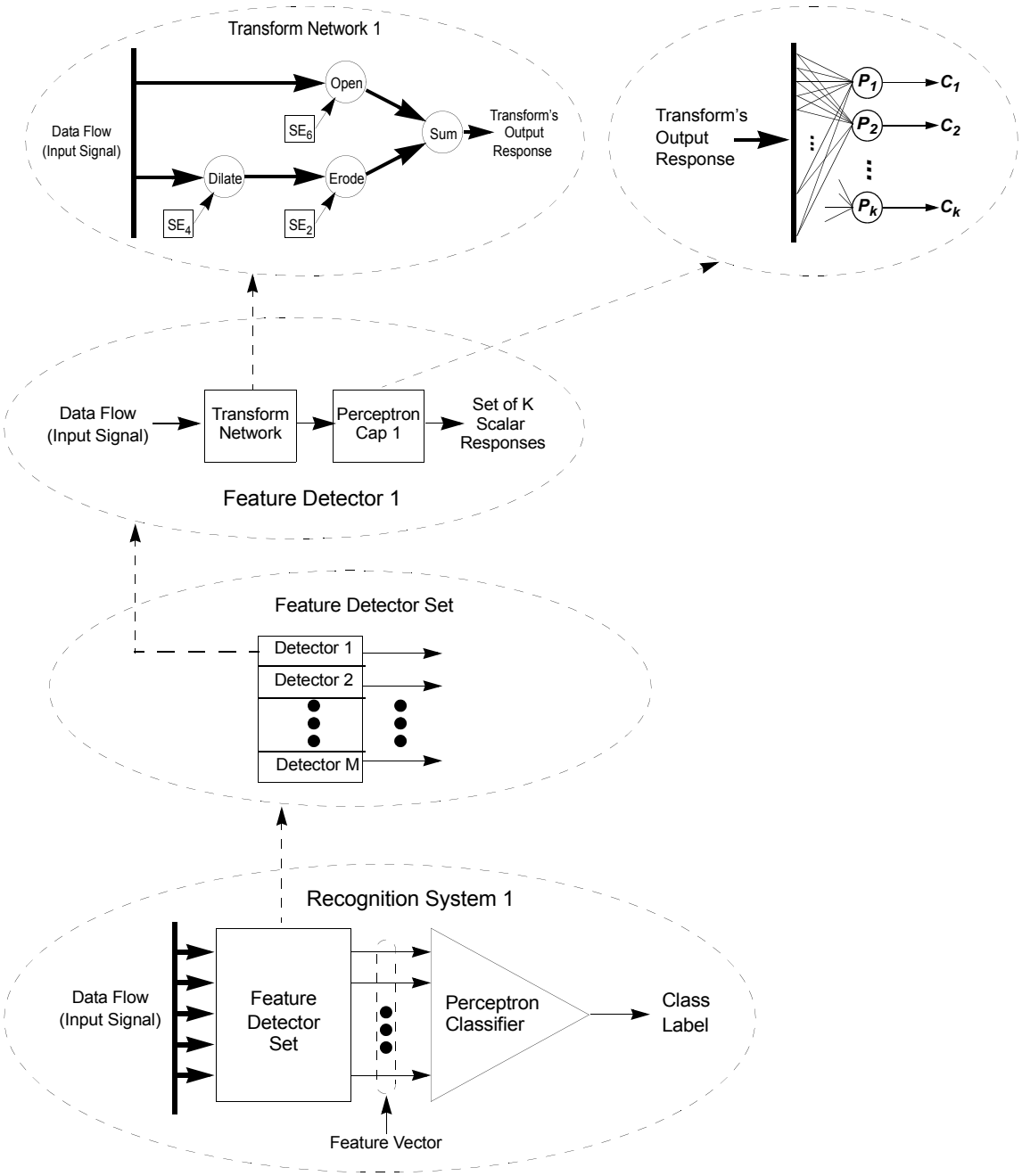


Figure 11. HELPR Representation used for HHR

each feature vector. A feature detector is composed of two components: a transformation network and a capping mechanism.

4.1.1.1 Transformation Network

Each transformation network is composed of morphological, arithmetic, and conditional operations that alter the input signal in an attempt to enhance the most-discriminating regions of the input while suppressing noise. When a transformation network processes a digitized input signal composed of n elements, the output is an n-element response vector. The output of a transform network is then capped to reduce the dimensionality of the signal. The key to evolving a robust recognition system is to synthesize transform networks within the feature detectors that facilitate the separation of targets into appropriate classes.

Mathematical morphology is a technique for probing the structure of signals or images using set theoretic operations [30, 51]. Each morphological operation is a signal-to-signal transformation that applies a probe-like pattern, referred to commonly as a structuring element (SE), to an input signal to produce an output signal. By selecting the correct algebraic form and structuring elements, specific objects can be isolated or enhanced. An example of a transformation applied to a few HRR signatures is shown in Figure 12. The expression, (Op (* (BaC I Bar2,4) (BaO (Cl I Bar2) Ball3,5) < 10) Ball3), consists of four morphological operations and one arithmetic operation. The Figure 12 illustrates in a step-by-step fashion how a transform alters a set of signals to separate one class of targets from two other classes. Two samples (rows) of signals from three different classes (columns) are shown next to each operation in the figure.

Two basic types of morphological operations are used: openings and closings. A closing operation fills in the gaps between peaks, where the size and shape of the SE defines which peaks get merged. Opening removes small portions of the signal, where again the size and shape of the SE dictate which characteristics of the signal get removed. A thorough discussion of the operation can be found in [30, 51].

The other two operations shown in this example are band operations, which compute the difference between two applications of an operation with the same structuring element at two different scales. This operation tends to isolate peaks or valleys in the signals having a size between a particular range. Two examples are shown in the figure, a band closing with a bar at scales 2 and 4, and a band opening with a ball at scales 3 and 5. The remaining operation shown in the example is an arithmetic transform. In this example, the product of two transformed signals is formed. One signal is shifted relative to the other signal to allow the combination of information at two dis-

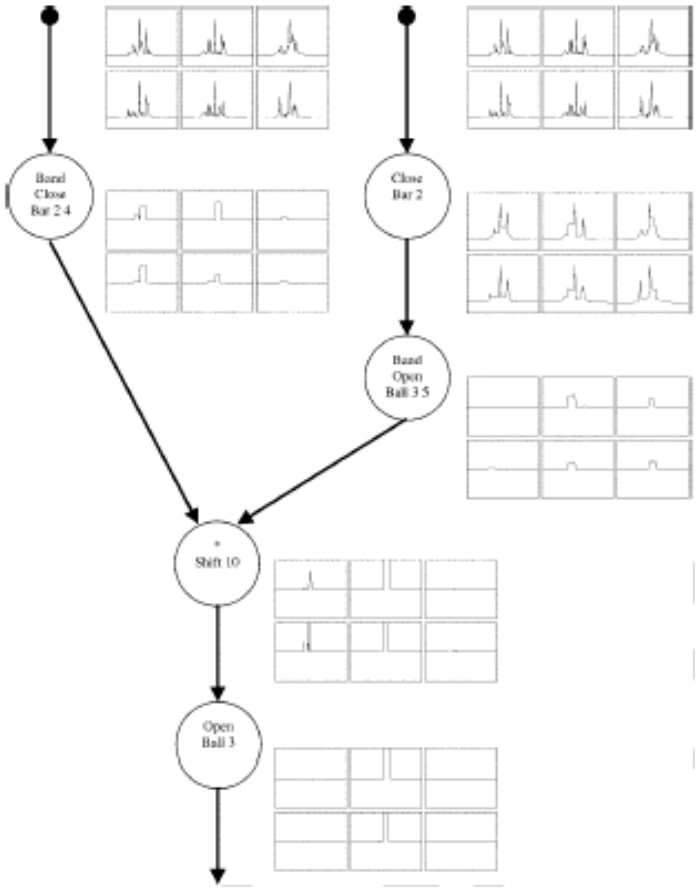


Figure 12. Sample Morpholoical Transform

tance positions in the transformed signals. While it was difficult to recognize the difference between the classes of signals in figure 2 before the transformation, the result is a discriminant that immediately separates one class from the other two.

Finding a transform and structuring element to separate two classes of signals is difficult, but finding a series of transformations that separate several classes of similar input signals is a demanding task even for an experienced morphological analyst. The need to explore and experiment with different combinations of operations, structuring elements, and parameters makes the task of synthesizing morphological expression well suited to evolutionary search. For this work, the full set of operations used in the evolutionary process include: erosion, dilation, opening, closing, band erosion, band dilation, band opening, band closing, and complement. In addition, four arithmetic operations were used: addition, subtraction, multiplication, and division. The structuring elements include various scales of bars, balls, and cones. Further examples of morphological

operations, library structuring elements, and the process of generating expressions are described in [70, 71].

The representation of the transform network supports both the definition of the form of an algebraic expression and the specification of parameters to adapt a specific expression to the nuances of a training data set. Thus, the task of evolving a transformation network can be viewed as a pair of concurrent search processes. One search explores an infinite space of algebraic networks to find the general framework of an expression that performs a coarse-grained transformation of an input signal to an alternative form while the second search adapts the parameters of the expression to fine tune the behavior of the transformation.

4.1.1.2 Capping Mechanism

The purpose of a transform cap is to reduce the dimensionality of a structured dataflow to a few scalar values. In the HRR application, the capping mechanism is composed of a single layer of linear perceptron nodes (p1, p2, … pK), where k is the number of classes. The HRR signal is a vector of scalars representing a set of range bins. The transform produces an output response vector. The input to each perceptron node is the response vector and the output is a scalar value. As illustrated in Figure 11, each detector has one perceptron for each of the k user-defined classes. The perceptron's weights are adjusted to use the transformed response vector to separate the training samples of one class of data from the remaining classes. Since transformations are relatively simple, the individual perceptron nodes do not act as highly accurate discriminant functions. Instead, the caps tend to split the output of transforms into subsets of related classes. The perceptron cap computes a weighted sum of outputs of a network of transforms. This allows the cap to rescale specific regions of a transform's output to increase separation between data samples from different classes. A bias term is introduced to allow the cap to shift the resultant value so that the response for a selected class is positive and other classes are negative. Notice the perceptron cap functions as a template or matched filter when applied to the transformed signal.

The use of a perceptron to cap a transform simplifies the evaluation of transforms because the value of the weight can be computed using a pseudo matrix inversion that adjusts the weights to minimize a least-square fit of the perceptron's output to training data, where the goal is to produce a value of +1 for a target class and -1 for all other classes. This simplification is only possi-

ble when the size of a transform's output response is relatively small. An alternative form of capping that evolves Gaussian kernels for extracting scalars from transform responses has also been investigated [12, 60]. This form is more appropriate when the dataflow contains large signals or two-dimensional images.

The sets of outputs from each detector are then passed to a second layer of perceptron nodes that serve as a classifier system (shown as "Perceptron Classifier" in Figure 11). Again, there is one perceptron node for each user-defined class. The ith node is trained to produce a +1 for a data sample drawn from the ith class and -1 for samples taken from all other classes.

4.1.2 Evolutionary Search

HELPR uses three different evolutionary techniques to alter the structure of the detector set contained in each recognition system. The parameters controlling specific properties of the transforms within a detector set are varied using aspects of evolutionary programming (EP), the structural form of the transformation is modified using genetic programming (GP), and the collection of detectors that form the basis of the feature extraction module are selected using a genetic algorithm (GA). These techniques are combined to exploit the strengths of each paradigm.

The overall flow of HELPR's evolutionary search process is shown in Figure 13. To begin, the user sets the size of the population of recognition systems. The user also specifies a minimum and maximum number of detectors to place into each recognition system to begin the evolutionary process. The transforms that define the initial detectors are generated at random. The user also sets limits on the size of the initial transforms. The limits on the number of detectors, the number of the transforms, and the number of operators in each transform is not critical because the evolutionary process adjusts the number of detectors and tunes the complexity of the transformations. After the transforms are created, caps are trained to separate the data into user-defined classes and the resultant outputs are passed to the classifier that assigns a fitness value to each recognition system based on its overall classification accuracy. Then the evolutionary learning cycle composed of repeated applications of reproduction with variation, evaluation, and selection begins.

The evolutionary process consists of the normal cycle of reproduction with variation, evaluation, and selection. The learning cycle continues for a fixed number of evolutionary cycles as

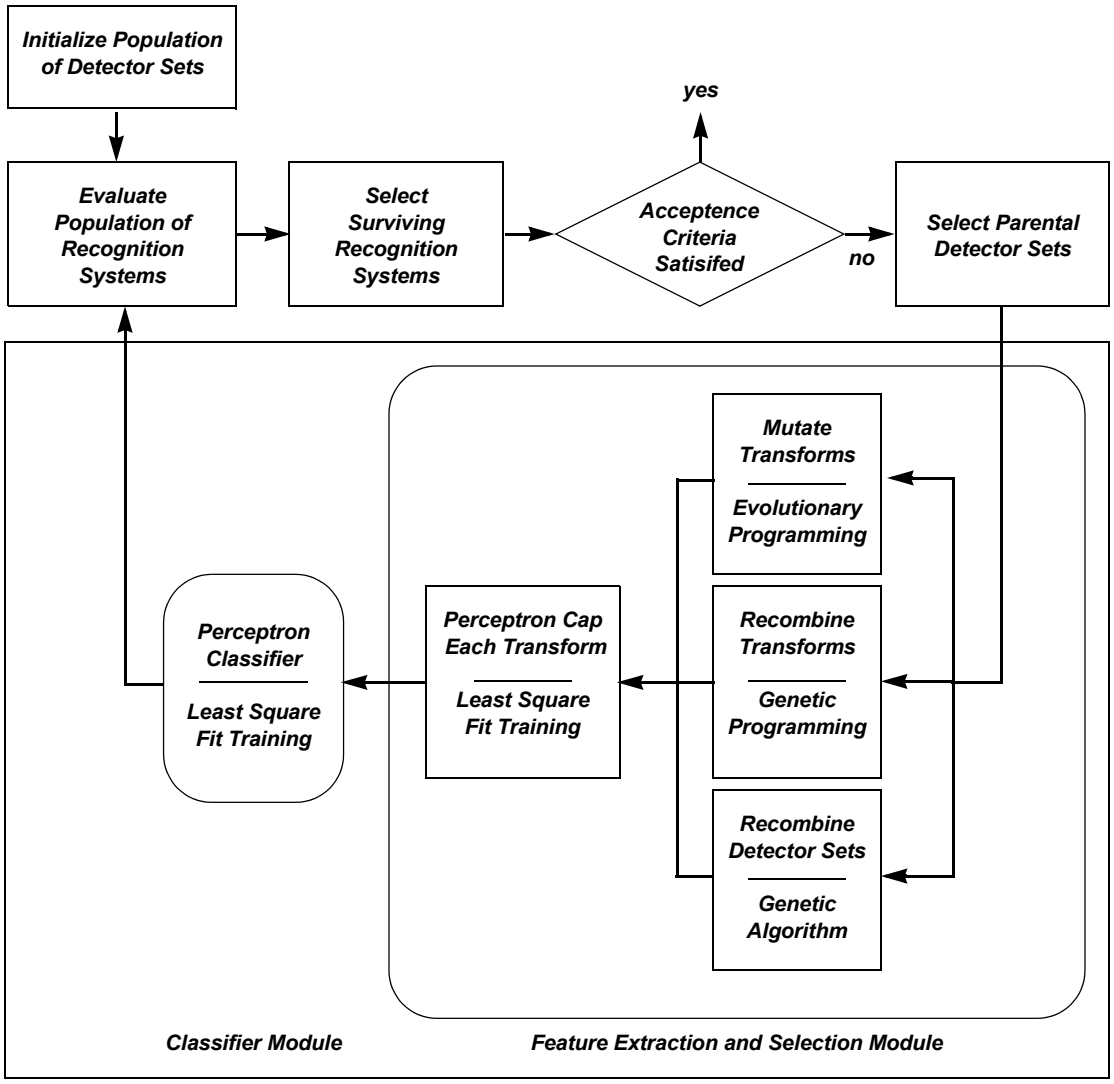


Figure 13. Learning Algorithm for HRR Targets

specified by the user. The parental recognition systems are chosen for reproduction using a scaled fitness proportional selection procedure [28, 29]. The selected parental units are then used to form offspring recognition systems. Pairs of parental systems are selected repeatedly and used to produce a user-specified number of offspring. The offspring are evaluated to determine their fitness, and then the parents and offspring compete for survival using a stochastic tournament selection procedure [21]. The details of reproduction, evaluation, and selection are described below.

Reproduction consists of producing variations of the genetic structure of parental detector sets. The heterogeneous representation of the genotype of a recognition system requires special

types of genetic operators. One approach considered was to apply different types of variations in phases. For example, complete detectors could be exchanged between parents in one phase, then transformation could be recombined in a second phase, and the parameters of the transforms could be mutated in the final phase. This approach imposes an artificial restriction on the way the genetic structure evolves and induces artifacts in the evolutionary process that limit the search process to a subset of the full range of genetic structures. The approach adopted in HELPR is to blend the variations so that during reproduction the genetic structure of the offspring are formed using different types of genetic variations. To control the amount of each type of variation, a probability distribution is introduced that assigns a probability to selecting each type of variation. There is distribution that controls the probability of a mutation, recombination, or a chromosomal aberration such as the addition or deletion of a gene (whole detector). When recombination is selected it can lead to the exchange of two complete detectors or exchange of pieces of transform networks. Each type of recombination has a specific probability that is control by a second user-defined probability distribution. The third probability distribution controls the amount of different types of mutation applied to individual detectors. These types of mutations include the addition/deletion of nodes in the networks, variations to the parameters associated with network nodes, joining networks under the control of a newly selected operator.

To begin the learning experiment, a population of pattern recognition systems is generated. Each system is initialized with a small set of randomly generated detectors consisting of a transform containing a few operators and a perceptron cap. The accuracy of each detector is evaluated by training a single layer of perceptrons to classify the training data set.

The evolutionary cycle begins by sampling a PDF that defines the probability of each type of variation. For these experiments, the probabilities were determined experimentally. The first decision is to select the basic type of variation: mutation or recombination. If mutation is selected, a single parental detector set is selected at random from the population. A copy of the parental detector set is formed and a PDF is sampled to determine the type of mutation to apply. Mutations consist of adding randomly generated detectors, deleting detectors, or varying the parameters and structure of existing detectors. A probability distribution is sampled to select the type of mutation to apply to each node in the network. The types of mutations include substitution of one operator or structuring element with another or a small variation in the parameters associated with the

operator. When parameters are mutated, an EP-like process is used to constrain the amount of variation introduced to promote gradual change in the offspring.

If the initial choice for the type of variation is recombination, two parental detector sets are selected at random from the population. The detectors are then recombined using a uniform crossover to form a pair of offspring detector sets as shown in Figure 14. This process is similar to a standard GA operation except that entire detectors are exchanged. As detectors are selected for placement in the offspring, they are subjected to a GP process that involves recombination of the their transform networks. This involves pruning a randomly selected subtree from each pair of corresponding transformations and grafting the subtrees back into the opposing transformation. The probability of each type of recombination is determined using a user-defined PDF.

After the offspring detector set is formed, a portion of the training data set is processed. This produces a set of transformed signals. Perceptron caps are generated to reduce the transformed signals to scalar outputs. One cap is trained for each class in the training data set. The goal is to adjust the perceptron weights to output a +1 for data samples from the selected class and -1 for samples from the remaining classes. If the training set contains K classes, then K perceptrons are trained for each transform. The outputs of these caps serve as the primitive features for the evolving recognition system. The capping process is repeated for each transform in the offspring resulting in the formation of a feature vector. The final step in forming a recognition system is to train a second layer of perceptrons to classify each data sample. One perceptron is assigned to each class in the training data. Each perceptron re-weights the feature vectors in an attempt to maximize its output for its designated class. A second subset of training data is then classified and the recognition accuracy is used to define the fitness of the newly formed offspring recognition system.

A population of size n is doubled during reproduction to size 2n and then reduced to n individuals using stochastic tournament selection. The value of n in our experiments is usually very small due to the high computational costs associated with evaluating complete pattern recognition systems. The small population size, coupled with the use of a selection technique that picks survivors from the combined set of parents and offspring led to the choice of a relatively small reproduction factor of 2 (n -> 2n) to alleviate problems with premature convergence. In addition,

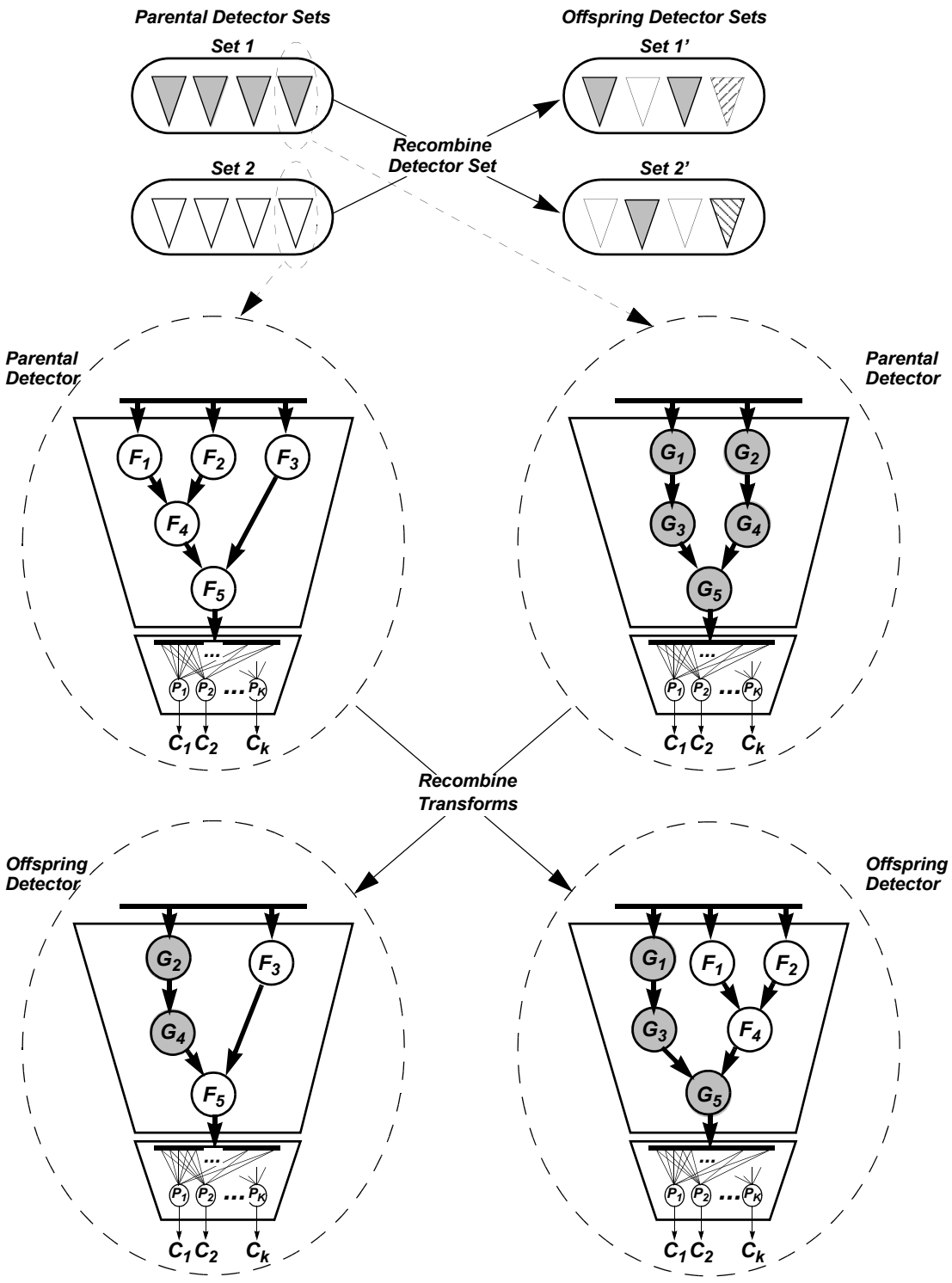


Figure 14. Sample Detector Recombination

the size of the tournament is scaled by the average fitness of the parental population. At the beginning of the evolutionary process when fitness is low, the tournaments are restricted to a few indi-

viduals, but as the average fitness increases the size of the tournament grows and the selective pressure increases proportionately. After the survivors are selected, the evolutionary cycle begins again. This process continues for a user-specified number of generations or until the population converges to a set of individuals with perfect training accuracy.

4.1.3 Experimental Results

To demonstrate how HELPR generates pattern recognition systems, the results of a target recognition task in HRR radar are presented. Specifically, the problem is to classify a set of airborne targets from their radar signatures. For this experiment, a database containing 6,426 sample radar signatures of airborne targets was processed. Each radar signature is one view of a target at a specific azimuth and elevation. The selected data set contains six targets at azimuths that range from -25° to +25° and elevations that range from -20° to 0° in increments of 1°. Thus, there are 1,071 (51*21=1071) samples of each target in the data set. Sample signatures are displayed in Figure 15. These signatures correspond to six different types of aircraft (columns) and they are shown at pose angles that vary by 1° increments in azimuth (rows). Each signature consists of 128 range bins containing values in the interval [-128,+128]. The signals are shifted so that bin 63 contains the maximum value. Looking down the column of data it is easy to see there are characteristic peaks and valleys in each target that persist through a few degrees of change in azimuth, but then disappear rapidly. Also note the similarity in the signatures between targets, which makes the classification task quite difficult. The value in each range bin is directly proportional to amount of energy received in a small unique time window. Typically this energy is reflected from surface elements within a limited range of distances from the radar receiver, however, this interpretation is complicated by multibounces of the reflected radar energy. Thus, the time distribution of the reflected energy is sensitive to small changes in the target's pose.

The signatures were divided into a training set containing 25% of the targets at randomly selected pose angles (6 targets * 267 poses = 1,602 samples) and the 75% of the data was placed in a test/validation set (6 targets * 804 poses = 4,824 samples). The training set is further divided into primary and secondary training set each containing approximately half of the training poses. The purpose of the primary training set is to train the perceptron classifier, while the secondary training set is used to assess the classification accuracy of the evolved recognition systems. It is

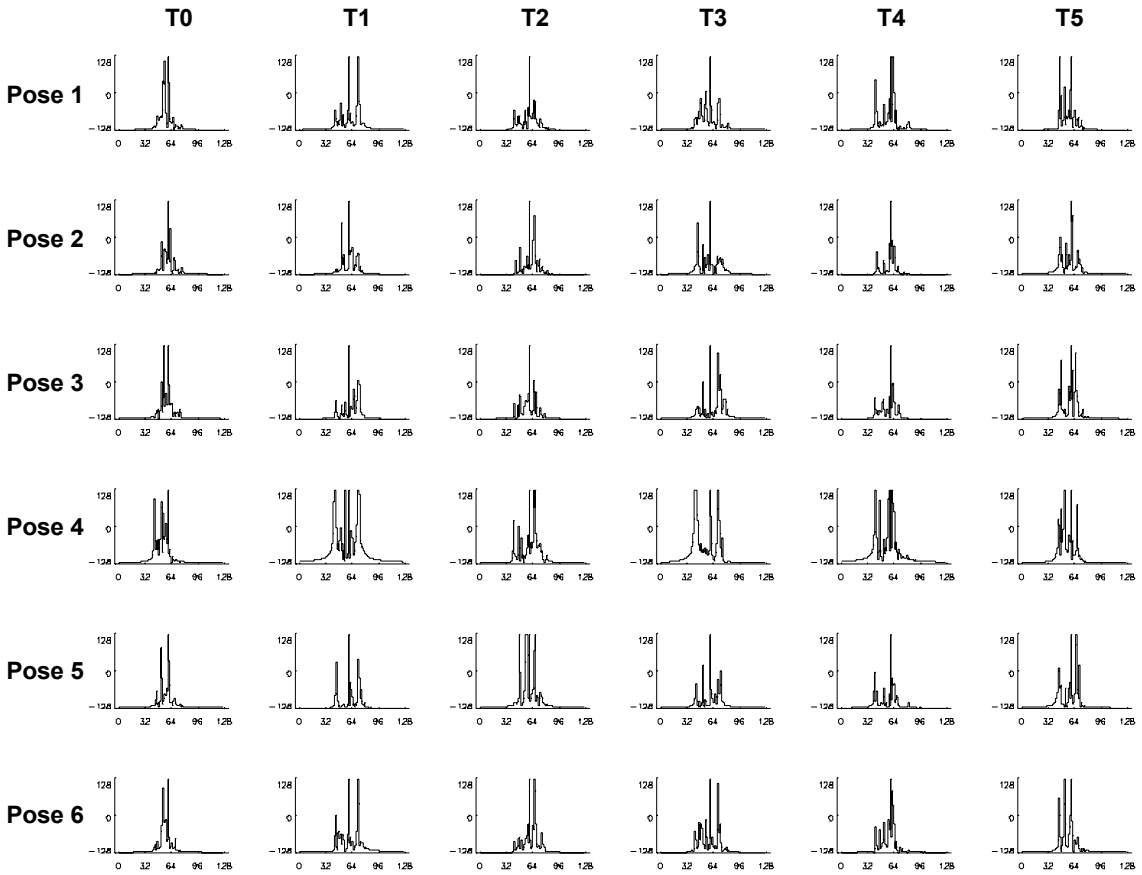


Figure 15. Sample HRR Signatures

the accuracy on the secondary training set that serves as the fitness measure. Prior to evaluating the test set, the two training sets are recombined and used to adjust the weights of the perceptron caps and classifier. This allows the HELPR system to take full advantage of all of the training data in formulating the final classification system.

The experiment consists of creating an initial population of ten recognition systems. Each recognition system contains two detectors constructed with one to three randomly selected operators. The population was then allowed to evolve for 300 generations and summary statistics were gathered every tenth cycle. This experiment was repeated ten times. For each replicate, the data set was re-sampled to obtain a different 25% of the target poses for training. The recognition system achieving the highest accuracy on the training data in the final evolutionary cycle was selected as the final product of the HELPR system.

The average accuracy of the ten best recognition systems on the combined training set (full 25% of the training data) and test set is displayed in Figure 16 by generation. The typical

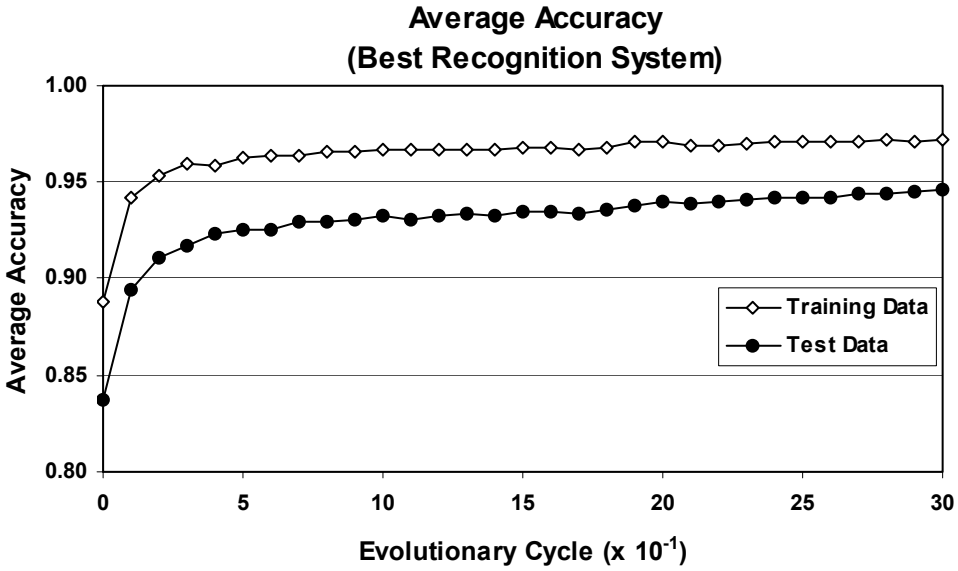


Figure 16. Average Recognition Accuracy By Cycle

training score begins around 85% accuracy. This relatively favorable beginning is the result of the perceptron classifier. When the raw signals were processed using just a network of perceptrons, accuracies were typically around 85% on the training data (see Table 2 which is discussed later). The shape of the learning curve suggests that bulk of the improvements occurred during the first 50 cycles. Most of this improvement can be attributed to an increase in the number of feature detectors in the best recognition systems as shown in Figure 17. After 50 cycles, the number of detectors leveled off at around 17, but performance continued to rise gradually for the remainder of the experiment. This can be attributed to refinement of the structure of the individual features that occurred later in the evolutionary process.

To determine whether the HELPR system was actually generating good recognition systems, a baseline performance was established using three techniques to classify the data. The simplest technique tested was a classifier consisting of a single-layer network of linear perceptrons. One perceptron was trained to discriminate each target class from the remaining classes. The second technique involved the use of a nearest neighbor classifier. For this technique, each of the sig-

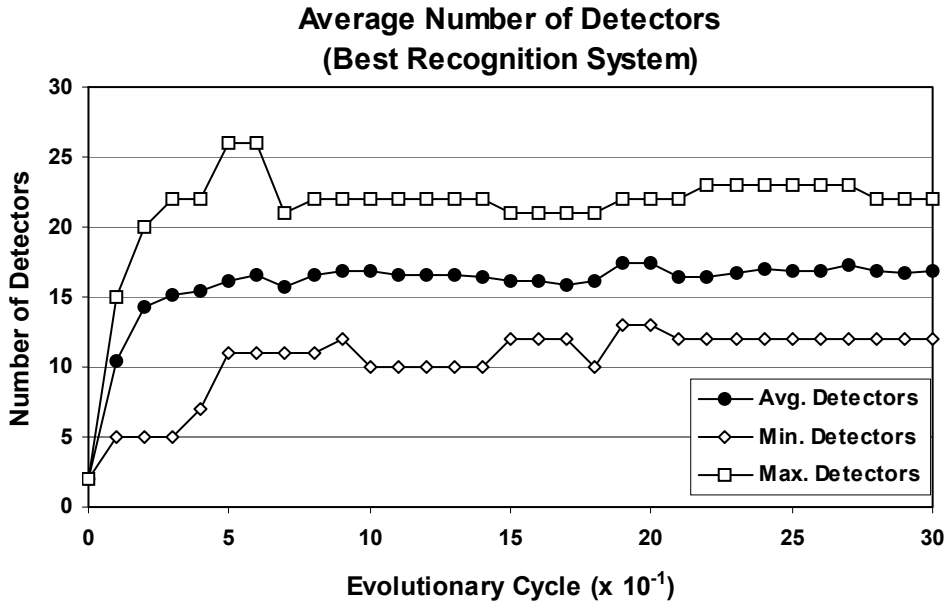


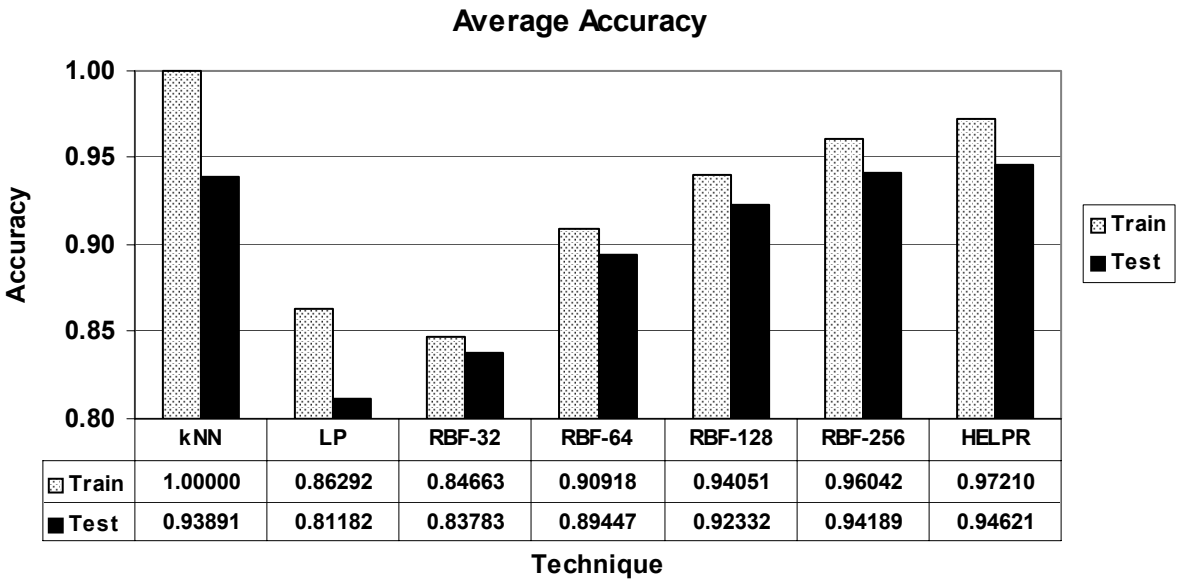
Figure 17. Average Complexity By Cycle

natures in the training set was used to define a prototype or cluster center for the classifier. The radius of attraction of each prototype was defined as the distance from the cluster center to a point half way between the farthest target correctly classified before reaching the closest misclassified target. Using this procedure, each prototype responds correctly to as many targets in the training data set as possible without producing any errors. The third technique evaluated was a radial basis network, which was constructed using a specific number of radial basis functions. Half of the basis functions were constructed by selecting random signals from the training data to serve as seeds for the cluster centers defining the positions of the basis functions. The remaining cluster centers were selected throughout the search space determined by the range of data in the training signatures. A Hardy multi-quadratic function was used to control the extent of the basis functions.

The network of perceptrons used in the first baseline technique forms a linear classifier. If the data are linearly separable, a single layer of perceptron will accurately classify the data. The nearest-neighbor classifier used in the second baseline technique defines a nonlinear classifier with the cluster centers serving as features. In the final baseline technique, the radial basis network is also a nonlinear classifier, which is a cross between a nearest neighbor classifier and a single layer network of perceptrons. The basis functions are essentially equivalent to the nearest

neighbor classifiers' cluster centers with a Gaussian-like envelope surrounding the center that defines a weight for the strength of attraction to each target. The outputs of the cluster centers are aggregated using a single layer of perceptrons.

Each of the baseline techniques was applied to exactly the same data sets that were processed using HEPLR. This produced ten scores for each technique. The summary statistics for the baseline techniques and the HELPR generated recognition systems are shown in Figure 18 and Table 2. The nearest-neighbor classifier achieved the best training accuracy. This is not surprising



	kNN	LP	RBF-32	RBF-64	RBF-128	RBF-256	HELPR
Train	1.00000	0.86292	0.84663	0.90918	0.94051	0.96042	0.97210
Test	0.93891	0.81182	0.83783	0.89447	0.92332	0.94189	0.94621

Figure 18. Baseline Comparison

since every training sample was a prototype in a nearest-neighbor classifier. Ignoring this anom-

Table 2: Summary Statistics

Technique	Training Set				Test Set			
	Avg	**Std**	**Min**	**Max**	**Avg**	**Std**	**Min**	**Max**
Nearest Neighbor	1.0000	0.0000	1.0000	1.0000	0.9389	0.0029	0.9366	0.9453
Perceptron	0.8629	0.0061	0.8496	0.8708	0.8118	0.0072	0.8010	0.8246
Radial Basis 32	0.8466	0.0184	0.8177	0.8727	0.8378	0.0071	0.8286	0.8487
Radial Basis 64	0.9092	0.0136	0.8814	0.9295	0.8945	0.0084	0.8835	0.9080
Radial Basis 128	0.9405	0.0077	0.9214	0.9488	0.9233	0.0059	0.9154	0.9316
Radial Basis 256	0.9604	0.0052	0.9538	0.9694	0.9419	0.0045	0.9335	0.9478
HELPR	0.9721	0.0122	0.9457	0.9825	0.9462	0.0124	0.9202	0.9608

aly, all the techniques produced reasonable average training scores. The HELPR generated system produces the highest average score of 97.2% accuracy. The accuracy on the independent test set is more important. All the techniques show a drop off in performance between the training and test sets but HELPR's recognition system achieves slightly better accuracy than any of the baseline techniques. HELPR's nearest competitor is a radial basis network using 256 basis functions. The difference in accuracy between the HELPR's recognition system and the radial basis network is less than 0.5%, which is not statistically significant. What is significant is that the HELPR's recognition system uses 17 feature detectors on average while the radial basis network uses 256 features (basis functions). Although the basis functions are simpler than the features generated by HELPR, the basis functions must be stored while HELPR's features are computed on the fly. Finding systems that use a small number of features to solve the problem is important from an operational point of view. These experiments involve recognition of aircraft from a head-on pose (25). Assuming features are not reusable for different poses, a recognition system that accurately classifies targets in a full 360 viewing volume would require an order of magnitude more features. In addition, as the number of target classes increase or the dimensionality of the problem increases (1D->2D->3D), it is very likely that still more features will be required to maintain high

recognition rates. Consequently, systems like HELPR that are capable of synthesizing accurate recognition systems using a small set of carefully crafted features have an advantage over systems that produce very large sets of simple features.

The learning curve showing the accuracy of the best recognition system in the population for one replicate of the experiment is shown in Figure 19. This run was allowed to continue to

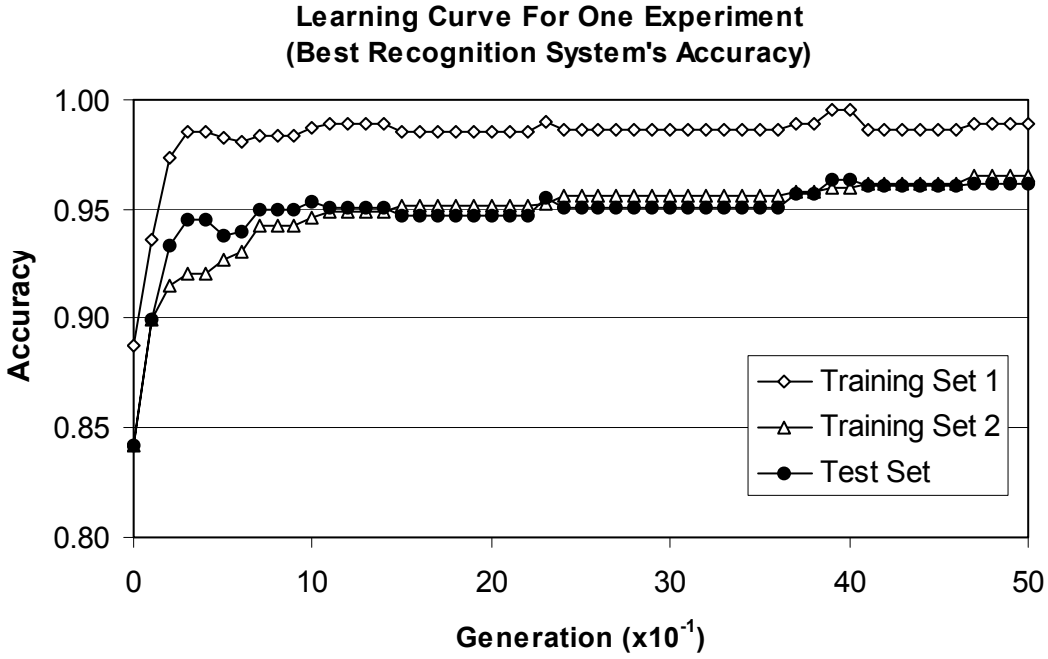


Figure 19. Learning Curves for One Replicate

evolve for an additional 200 learning cycles beyond the original 300 cycles used to generate the results reported in Table 3. For every cycle, the accuracy of the recognition system with the highest score on training set 2 was recorded. A small jump in performance occurred around cycle #375 suggesting that HELPR had not converged to a final solution. If the runs were allowed to continue beyond 300 cycles, additional gains in performance could have been achieved. At the end of 300 cycles, the HELPR recognition system declared a label for 4702 out of 4824 targets (declaration rate of 97.47%) and classified 4603 targets correctly (raw accuracy of 95.42% with accuracy on declared targets of 97.90%). After 200 additional cycles, this score changed to 4701 targets declared with 4633 classified correctly producing a raw test accuracy of 96.04% and an accuracy of 98.55% on declared targets.

Table 3: Accuracy of Best Solution

Technique	Raw Training Accuracy	Raw Test Accuracy	Correct Test	Declared Test	Correct Over Declared
HELPR	0.9825	0.9604	4633	4701	0.9855
Nearest Neighbor	1.0000	0.9366	4530	4723	0.9591
Perceptron	0.8600	0.8147	3930	4224	0.9303
Radial Basis-32	0.8177	0.82866	3997	4294	0.9308
Radial Basis-64	0.8814	0.8835	4262	4485	0.9502
Radial Basis-128	0.9213	0.9225	4450	4591	0.9692
Radial Basis-256	0.9544	0.9414	4569	4672	0.9779

Figure 19 also shows how the two-level training set was used to guide the evolutionary learning process. Training set 1 was used to adjust the weights for the transform network caps while training set 2 was used to score the evolving recognition systems. Notice the accuracy achieved on training set 2 is a good predictor of a recognition system's performance on the independent test set. This is not as obvious when the two training sets are combined (raw training score) and used to adjust the transform caps prior to processing the validation set. The accuracy on the combined training sets tends to overestimate test set performance by a few percent. In Table 2, the combined training and test set scores are reported for the best recognition system and compared to the performances achieved using the three baseline techniques. HELPR's recognition system achieves the best raw test set accuracy (i.e., ratio of targets classified correctly to total targets) of 96.04% among all the techniques. It also yields 98.55% classification accuracy on declared targets suggesting that the evolved recognition systems correctly labeled targets with a high level of confidence, which is very important in military applications.

In most pattern recognition problems, it is difficult to determine the relationship among the data samples within a given class. The HRR data set presents a somewhat unique opportunity to relate recognition errors to user-defined labels. The grid in Figure 20 shows the section of the

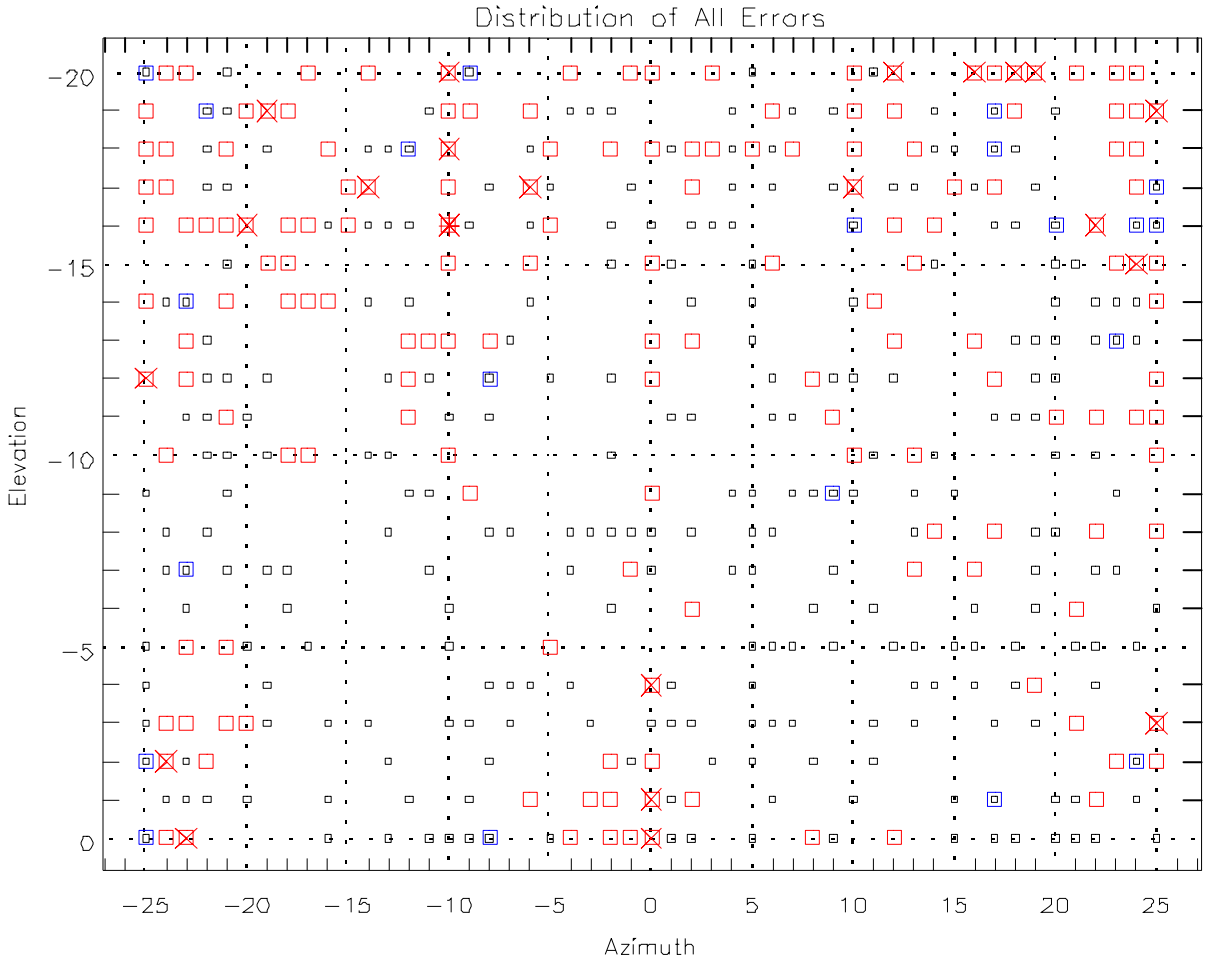


Figure 20. Errors By Pose

view window that defines the distribution of target pose angles used in these experiments. The azimuth range from -25 to +25 and elevations range from -20 to 0 producing 1071 poses. The small squares represent the randomly selected poses used to define the training data. The large squares mark poses where the evolve recognition system misclassified one of the six targets. A large square surrounding a small square is a training error while a large open square represents test set error. An X placed over a large square indicates two targets were misclassified at the marked location and an asterisk indicates three errors. Note there is only one asterisk in this grid at elevation -16 and azimuth of -10. To interpret these results, visualize staring head-on at the nose of an aircraft. This corresponds to a pose angle of 0 azimuth and 0 elevation. As the azimuth increases or deceases, more of the side of aircraft becomes visible to the radar sensor. As the ele-

vation changes, more of the surface of the wings becomes visible. There is a small pocket of errors that occurs near 0 azimuth and 0 elevation because this pose provides limited information to discriminate among certain aircraft. As the pose angle begins to increase or decrease the errors diminish. As the posed angle becomes very large, the errors increase again. Also note that there is symmetry to the distribution of errors with respect to poses at positive and negative azimuths, which is expected given the bilateral symmetry typically seen in aircraft.

In Figure 21, the distribution of errors is attributed to specific classes of targets. The

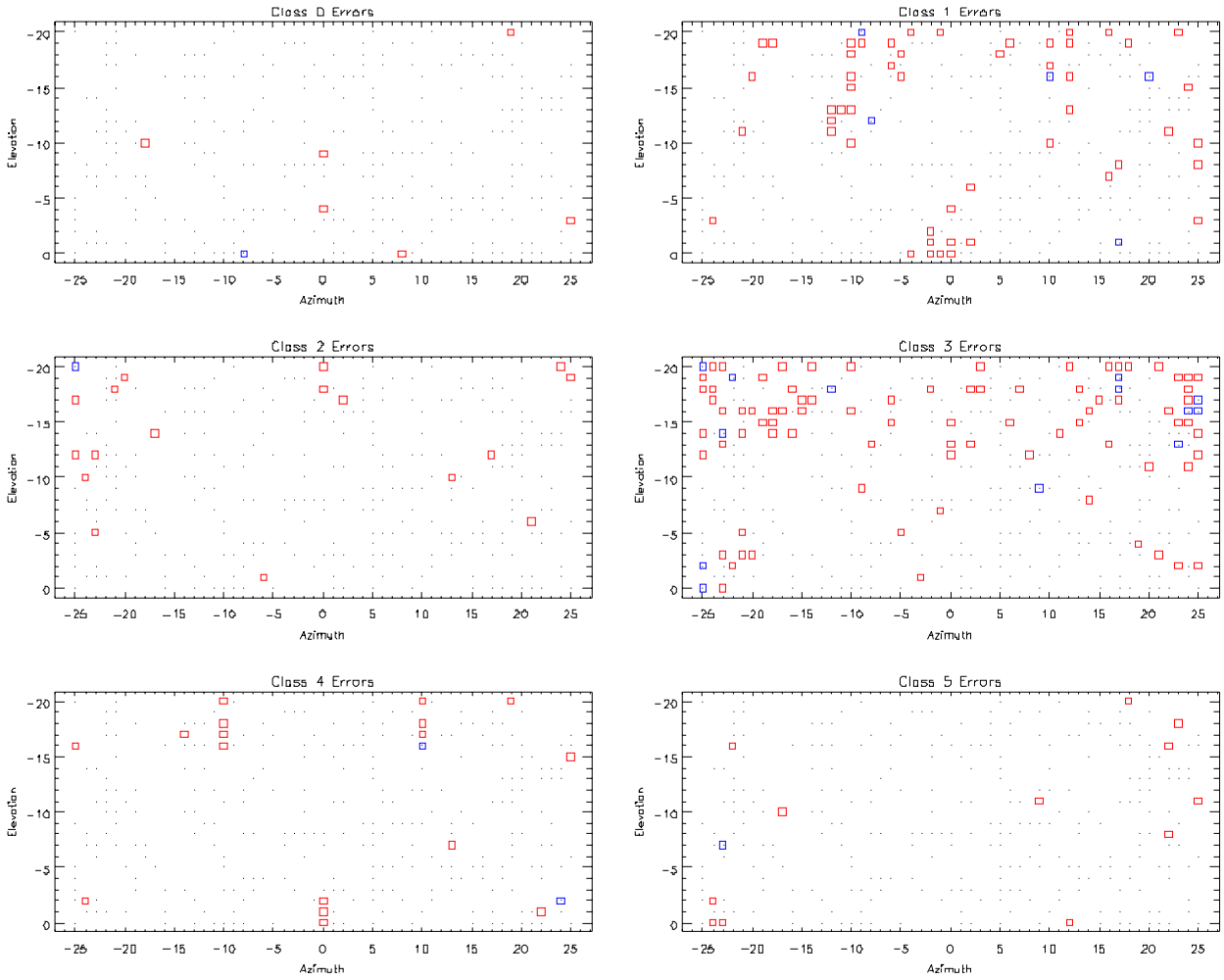


Figure 21. Errors By Pose and Target Class

majority of errors occur in classes 1 and 3. In many instances, these two classes have very similar signatures. Compare the signatures in the columns labeled T1 and T3 of Figure 15. The distribution and magnitude of peaks in signatures of targets 1 and 3 at specific pose angles tends to be

more similar than signatures in classes 0, 2, 4, and 5. This suggests that in future experiments an approach to increase overall recognition accuracy will be to remove the signatures of non-interfering classes from the training data, evolve features that separate the interfering classes and then incorporate these features back into an evolutionary process that synthesizes feature sets for the full data set.

The transforms that evolved during this specific run of the experiment are shown in figure 22. The complexity of the transforms is quite diverse. The simplest transform, T10, consists of an identity operator that simply passes the raw signal through to the capping processes. This amounts to incorporating a set of linear perceptron nodes into the solution. The baseline experiment demonstrates that a single layer of perceptrons applied to the raw inputs produces 81.47% accuracy on the test data. The remaining transforms work to further increase accuracy to the level of 96.04% on the test set. The effect of the transforms on a few sample HRR signatures is shown in Figure 23. The columns corresponds to six targets at the same pose angle (elevation -3, azimuth -7). The rows show the effect of applying the transforms given in Figure 22 to each target. The simplest morphological transform is T3, which closes the signals with a ball-shaped structuring element of size 2. This fills gaps between peaks that are smaller than the diameter of the ball. This is demonstrated clearly in the plot of the class 4 (C4) signal for transform T3. The two highest peaks that are close together are merged into one peak. At the same time, the three highest peaks in C1 that are farther apart than the diameter of the ball remain relatively unchanged. Notice how each transformation alters the input in some unique fashion. This amounts to restructuring the information in the original signal so that the caps then readily locate regions of the signals that discriminate among the targets.

Although the transforms shown in figure 22 appear to be very complex expressions, many of the sequences incorporate common terms. For example, transforms T0, T4, T7, T11, T14, and T17 all have the common term: Cl I Ball2. This term can be factored out, computed once, and then used in all the expressions. There are also larger terms such as (BaD (BaC I Ball2,8) Ball6,8) that appear in several terms ((T0, T11, and T15) that can be factored. In addition, there are redundant terms such as (Close (Close X Ball6) Ball6) that can be reduced to (Close X Ball6). Although these redundant terms can be removed during the evolutionary process, the performance of the system is degraded if they are removed too early in the evolutionary process. Such redundant

T0: (Di (Cl (* (BaC I $Ball_{0,4}$) (BaD (/ I (BaD (BaC I $Ball_{2,8}$) $Ball_{6,8}$) >15) $Ball_{6,8}$) >15) $Cone0.5_0$) $Cone2.0_0$)

T1: (BaC (Op (+ (Di I $Cone2.0_0$) (Er (BaD (BaD I $Ball_{6,8}$) $Ball_{6,8}$) $Cone0.5_8$) >50) $Cone2.0_5$) $Ball_{0,4}$)

T2: (Er (+ (BaD (BaD I $Bar1.0_{0,6}$) $Ball_{6,8}$) (BaC I $Cone1.0_{1,5}$) <58) $Cone0.5_8$)

T3: (Cl I $Ball_2$)

T4: (Er (+ I (Er (BaD (Cl (* (BaC I $Ball_{0,4}$) (BaD (BaD (BaE I $Cone2.0_{0,7}$) $Ball_{6,8}$) $Ball_{6,8}$) >15) $Cone0.5_0$) $Ball_{6,8}$) $Cone0.5_8$) >50) $Cone1.0_0$)

T5: (BaC (Cl I $Ball_2$) $Ball_{0,4}$)

T6: (Di (Cl (Er I $Cone0.5_8$) $Cone0.5_2$) $Cone2.0_0$)

T7: (BaC (+ (+ (BaC I $Ball_{0,4}$) (BaC I $Ball_{0,4}$) >50) (BaC (Op (+ (Di I $Cone2.0_0$) (Er (+ (Di I $Cone2.0_0$) (Er (BaD (BaD I $Ball_{6,8}$) $Ball_{6,8}$) $Cone0.5_8$) >50) $Cone0.5_8$) >50) $Cone2.0_5$) $Ball_{0,4}$) >22) $Ball_{0,4}$)

T8: (* I (BaD (BaC (+ (Di I $Cone2.0_0$) (Er (+ (BaC (BaD (BaD I $Bar1.0_{0,6}$) $Ball_{6,8}$) $Cone1.0_{1,5}$) (BaD (BaC I $Ball_{2,8}$) $Ball_{6,8}$) >50) $Cone0.5_8$) >50) $Ball_{2,8}$) $Ball_{6,8}$) >15)

T9: (* I (/ I (BaD I $Ball_{7,8}$) >15) >31)

T10: I

T11: (Cl (+ (BaC I $Ball_{0,4}$) (Er (BaD (+ (+ (/ I (BaD (BaC I $Ball_{2,8}$) $Ball_{6,8}$) >15) I >50) (Op (/ I (BaD (BaC (BaC I $Cone1.0_{1,5}$) $Ball_{2,8}$) $Ball_{6,8}$) >15) $Cone1.0_5$) >22) $Cone1.0_{6,8}$) $Cone0.5_8$) >50) $Ball_5$)

T12: (- (+ (BaC I $Ball_{2,8}$) (Er I $Cone0.5_8$) >50) (BaD (BaC (Cl I $Ball_2$) $Ball_{0,4}$) $Ball_{6,8}$) <17)

T13: (BaE I $Cone2.0_{0,7}$)

T14: (+ (Op (+ (BaC I $Ball_{0,4}$) (BaD I $Ball_{6,8}$) >50) $Ball_8$) (Cl (BaC (Op (+ (Di I $Cone2.0_0$) (Er (BaD (BaD I $Ball_{6,8}$) $Ball_{6,8}$) $Cone0.5_8$) >50) $Cone2.0_5$) $Ball_{0,4}$) Ball 7) >48)

T15: (+ (/ I (BaD (BaC I $Ball_{2,8}$) $Ball_{6,8}$) >15) (Di (Di (Cl I $Cone0.5_2$) $Ball_0$) $Cone2.0_0$) >50)

T16: (Op (Di (- I (+ (BaC (BaC I $Cone1.0_{1,5}$) $Ball_{0,4}$) (+ (BaC I $Ball_{0,4}$) (BaD (BaC (Cl I $Ball_2$) $Ball_{0,1}$) $Ball_{7,8}$) >50) >50) >9) Cone2.0 2) $Bar1.0_0$)

T17: (Cl (Er (Di (Cl (BaD (- I (+ (BaC I $Ball_{0,4}$) I >50) >9) $Bar1.0_{0,6}$) $Cone0.5_2$) $Cone2.0_0$) $Ball_1$) $Cone2.0_4$)

T18: (Cl (Di I $Ball_4$) $Cone0.5_2$)

Figure 22. Sample Evolved Transforms

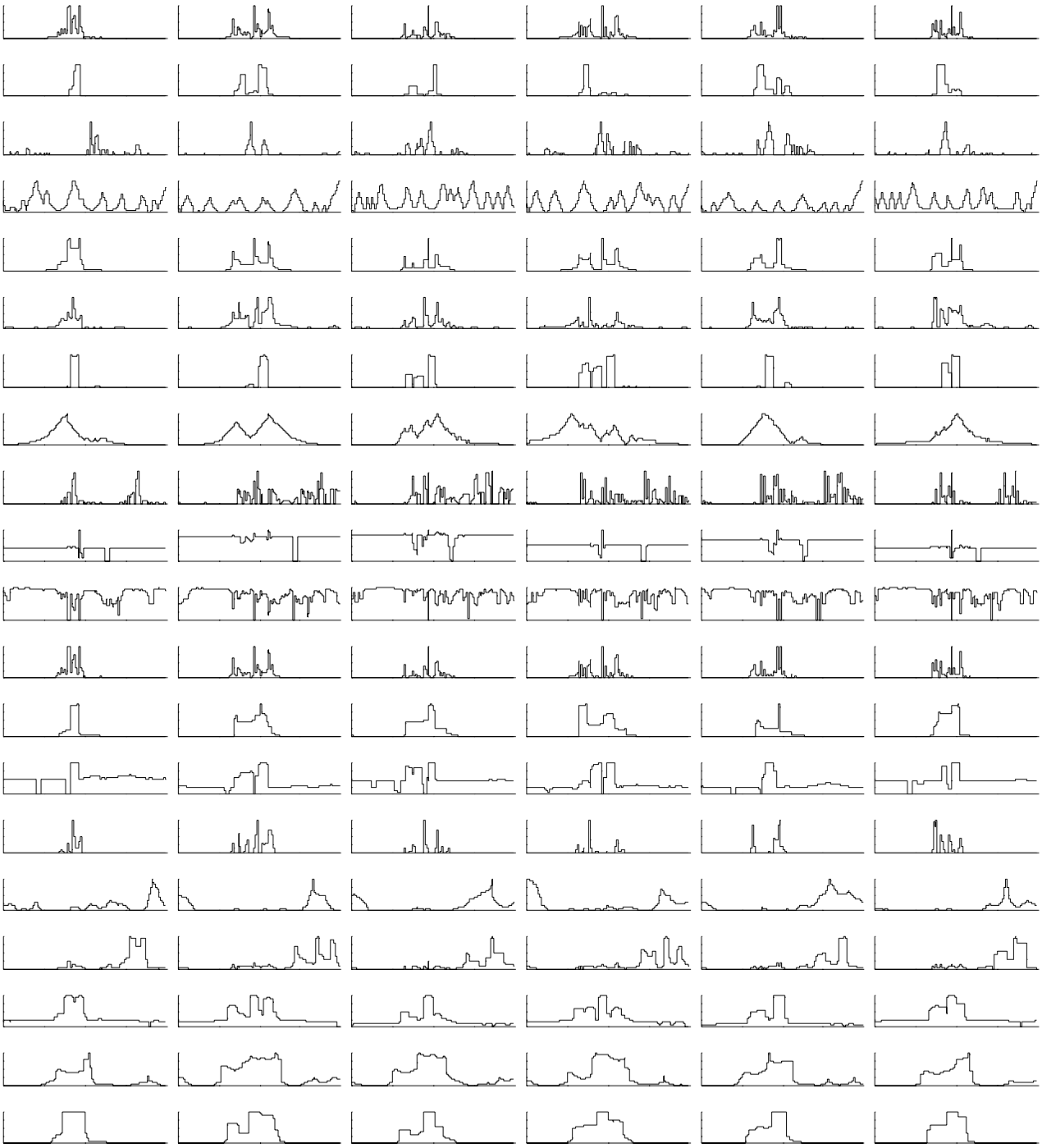


Figure 22. Sample Transformed HRR Signals

terms appear to provide a scratchpad for evolutionary experimentation in a fashion similar to introns and exons in biota. When the evolutionary process terminates, common terms can be factored out easily and redundant terms removed to simplify the transforms and produce a more computationally efficient recognition system.

In figure 23, the final response of the perceptron classifier is shown. The left column of

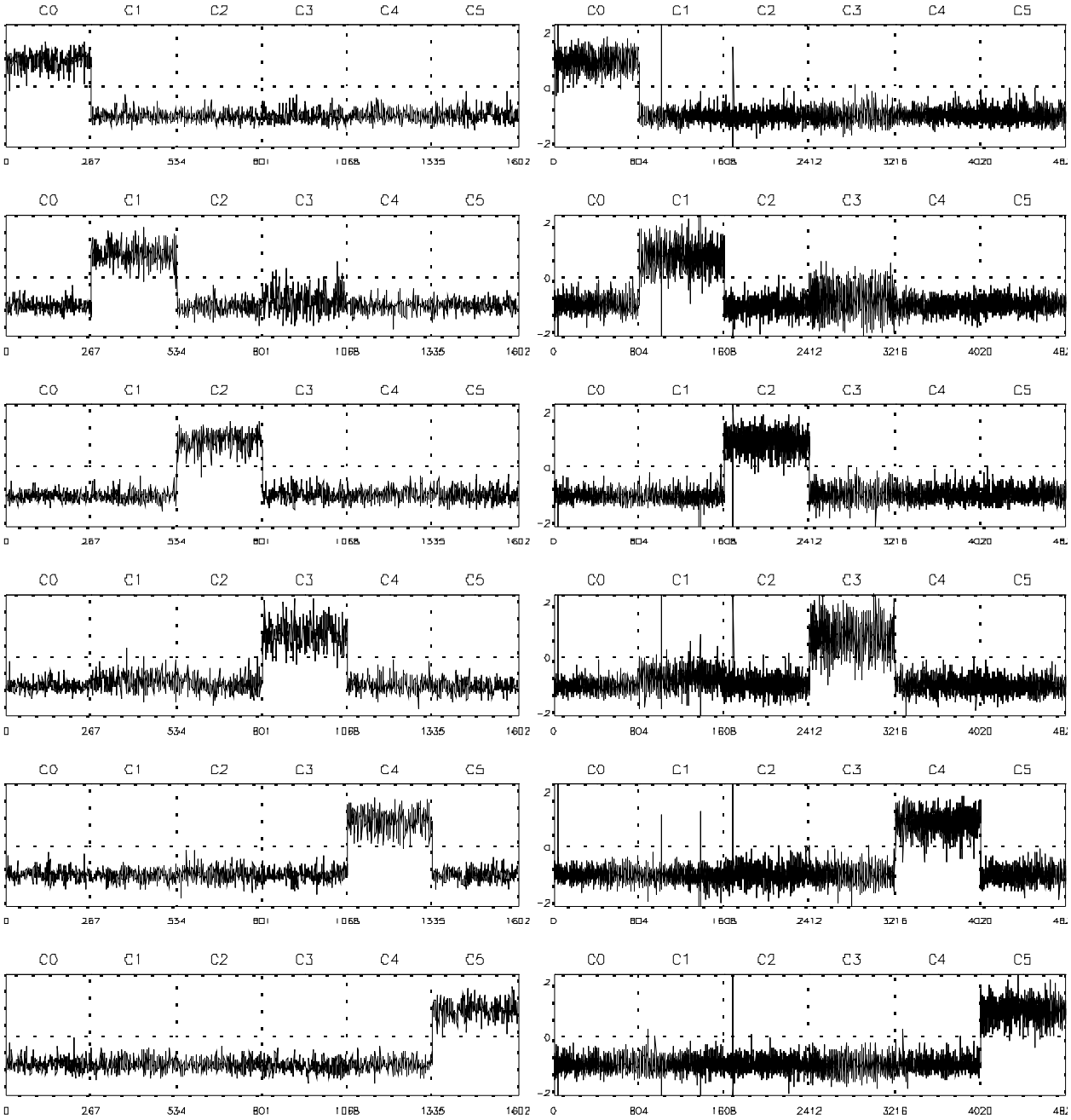


Figure 23. Sample Response Vectors

plots illustrates the response of the resulting recognition system to the training signal and the right column of plots represents the response to the test images. Notice the scales on the x-axis of the two columns of plot are different. The rows correspond to the individual perceptron discriminant

functions. Perceptron Pi is trained to produce a positive value for signals in class Ci and a negative response for signal from the remaining classes. A horizontal dotted line is included in each plot to help the reader locate the zero value. The vertical dotted lines mark the separations between classes. The responses demonstrate that the capped transformed signals produce relatively consistent responses across the each class. There is clearly more variation in the responses to the test signals, but overall the amount of variation between the training and test sets is consistent. The interference between classes 1 and 3 is obvious. There are also unusual isolated responses that tend to be off the scale of the plots. The spike among the test signals in class three is an example of this behavior. This is not a failure in the morphological operation. Analysis of the input signals revealed that there are few signals that appear to be highly distorted. This is due to a glint effect that occurs when the radar aligns with some surface of the target and energy is reflected back directly into the sensor. These signals have no discernible structure and appear to be random noise.

4.1.4 Discussion of HRR Results

The HELPR system synthesizes pattern recognition systems using a minimum amount of user interaction. The recognition systems that evolve use fewer features than systems formed using conventional techniques, yet achieve comparable or superior recognition accuracy. In head-to-head comparisons between HELPR-generated recognition systems, perceptron networks, nearest-neighbor classifiers, and radial basis networks, HELPR's best recognition system had a higher overall recognition rate in the majority of replicates of the experiments and consistently produced better classification accuracy for declared targets.

HELPR is designed to evolve complete recognition systems using raw data as opposed to preprocessed sets of features. As a result, the system uses a multifaceted representation to evolve several layers of structure to form a recognition system. The first layer transforms signals using a combination of arithmetic and morphology operators to reorganize the dataflow to simplify the extraction of discriminating features. The second phase then extracts numeric feature values. The system must select both the number and type of features to pass to the final stage of the system for classification. The final stage forms a single layer of linear discriminant functions to perform the classification.

Although the evolved features are attached to linear discriminant functions for classification, the features are not independent. One of the interesting aspects of the HELPR system is that each feature can participate in the classification of several targets. This is an important issue when the number of classes is large because the complexity of the system will not grow in direct proportion to the number of target classes. For example, one feature may be sensitive to a specific aircraft wing profile. If several aircraft share this type of wing, the feature can contribute to separating targets with similar characteristics from those without the characteristic. Analysis of the evolved features reveals that there are many shared sub-expressions within the terminal features supporting the claim that cooperating features emerge through the evolutionary process. In real applications, shared sub-expressions can be computed once and then used by all the features detectors that incorporate those sub-expressions. This will increase the efficiency of the delivery recognition systems.

HELPR begins with a few random generated transforms and features and continues to elaborate on the existing features while adding additional features to improve classification accuracy. Although there are a few intermediate performance measures used to eliminate obvious nonviable features, the overall process is driven by terminal recognition accuracy eliminating the problems associated with using intermediate performances to develop solutions to complex nonlinear problems.

One of the unique aspects of the HELPR system is the use of morphological operators in the first stage of the system. By simply changing the dimension of structuring elements morphological operators are capable of operating on 1-D signals, 2-D images, and 3-D solid models. In general, morphological operators can be extended to operate on data in any dimension. Consequently, the HELPR system scales readily to tasks in a variety of problem domains. Various aspects of HELPR have been applied to problems involving 2-D data including optical character recognition, industrial inspection, and medical image analysis.

The next stage in the development of the HELPR system will explore the use of alternative capping mechanisms to perform feature extraction. The capping used in this experiment is based on a perceptron that weights every component of a transformed signal. This is only practical if the transformed signal is relatively small. Initial experiments using Gaussian probes to

locate regions of interest in the transformed signal or image appear promising. We have also developed a technique similar to the method described for extracting information from raw data that can extract complex features from a large set of pre-existing features. Combining the mechanisms described in this paper for synthesizing transform with a more general capping mechanism and a top-level system for extracting complex features will form a useful approach for automated design of the next generation of pattern recognition systems.

4.2 MSTAR Application

The HELPR system was also used to synthesize recognition systems for SAR data taken from the MSTAR data set. The representation used for HELPR was modified to handle images. This required changes to the preprocessing module to allow the use of 2D morphology and changes to the capping mechanism to reduce 2D responses to scalar values. In addition, a new performance measure was introduced that proved valuable for both classifying targets and predicting their pose.

4.2.1 Representation

The representation of HELPR shown in Figure 11 was modified. First the preprocessing module that used 1D morphology for HRR data was replaced with a module that used 2D morphology. Second, the Perceptron caps were replaced by a mechanism that used 2-dimensional kernels. Finally, the classifier was replaced with a KNN classifier. The HELPR principles are maintained in this modified architecture. Each recognition system is composed of a feature extraction module and a classification module. The feature extraction module applies a set of feature detectors to an input signal to form a feature vector. The set of feature detectors is viewed as a linear chromosome that defines the genotype of an individual recognition system. The classifier module then assigns a target label to each feature vector. A feature detector is composed of two components: a transformation network and a capping mechanism.

4.2.1.1 Transformation Network

As described in section 4.1.1, each transformation network is composed of morphological, arithmetic, and conditional operations that alter the input image in an attempt to enhance the

most-discriminating regions of the input while suppressing noise. When a transformation network processes an m x n input image, the output is an m x n response image. The output of a transform is capped to reduce the dimensionality of the input to a small set of scalar values. The key to evolving a robust recognition system is to synthesize transform networks within the feature detectors that facilitate the separation of targets into appropriate classes.

The morphological operators described in section 4.1.1.1 are altered from 1D to 2D. This results in the use of 2D structuring elements. This allows HELPR to explore greyscale SAR images using the same form of processing networks that were used for the HRR signals.

For this work, the full set of operations used in the evolutionary process include: erosion, dilation, opening, closing, band erosion, band dilation, band opening, band closing, and complement. In addition, four arithmetic operations were used: addition, subtraction, multiplication, and division. The structuring elements include various scales of bars, balls, and cones in 2D.

As in the HRR experiments, the representation of the transform network supports both the definition of the form of an algebraic expression and the specification of parameters to adapt a specific expression to the nuances of a training data set. Thus, the task of evolving a transformation network can be viewed as a pair of concurrent search processes. One search explores an infinite space of algebraic networks to find the general framework of an expression that performs a coarse-grained transformation of an input signal to an alternative form while the second search adapts the parameters of the expression to fine tune the behavior of the transformation.

4.2.1.2 Capping Mechanism

The purpose of a transform cap is to reduce the dimensionality of a structured dataflow to a few scalar values. In the SAR application, a cap is composed of a rectangular area at a specific position within the output of the transformed images. These area consists of a center location, a width, a height, and an operation. The rectangular area is positioned in the transformed image using the center location of the rectangle and the operation is applied to the pixel within the area of the rectangle. Several types of operations are available including minimum, maximum, average, variance, and median. These operator are designed to summarize the pixel values within the

rectangular area as a single scalar value. The sets of outputs from each detector are then used to form a feature vector that is passed to a KNN classifier.

4.2.2 Evolutionary Search

The same evolutionary search techniques described in section 4.1.2 are applied to the modified HELPR architecture. The only significant differences are occur with respect to the evolution of the reduction caps and the definition of the classifier. In this experiment, the caps are kernels. Each kernel specifies how to combine the pixel values in a selected region of a transformed image to form a scalar value. An evolutionary programming algorithm is used to mutate the position, size and type of operator used to define each kernel.

A new approach for evolving KNN classifiers was developed as part of the HELPR project that allows the evolve systems to cope with pose or ordered data. This approach attempts to form clusters in feature space that support accurate recognition, but also automatically orders the clusters in feature space based on cluster similarity. This allows the evolved recognition systems to maintain a level of robust performance under environmental variation as well as estimate the pose of a target.

In this new approach, the training images are processed using the set of transformations and scalar features extracted from the regions of interest defined by the caps. These extracted values form a real-valued feature vector that describes each training sample. These feature vectors are then used to parameterize a nearest neighbor classifier. Each training sample defines one prototype in the classifier. A radius of attraction is defined for each prototype that limits the prototype's range of influence. Since all the training samples are used as prototypes, each sample is attracted by the prototype that was defined by the sample. This presents a problem since there is no data left to evaluate the ability of the evolving pattern recognition system to generalize to new data sets. To solve this problem, we developed a new performance measure that rates a recognition system not based on raw accuracy, but based on how many training samples are attracted by neighboring prototypes. For example, assume the ith sample is used to define the ith prototype in the classifier. If sample (i+1) is attracted to prototype i, then we give the prototype a score of 2. If the (i-1) sample is also attracted to prototype i, we add 2 to the score of the prototype. If samples (i-2) and (i+2) are also attracted to prototype i, we add one point for each of these samples. Thus,

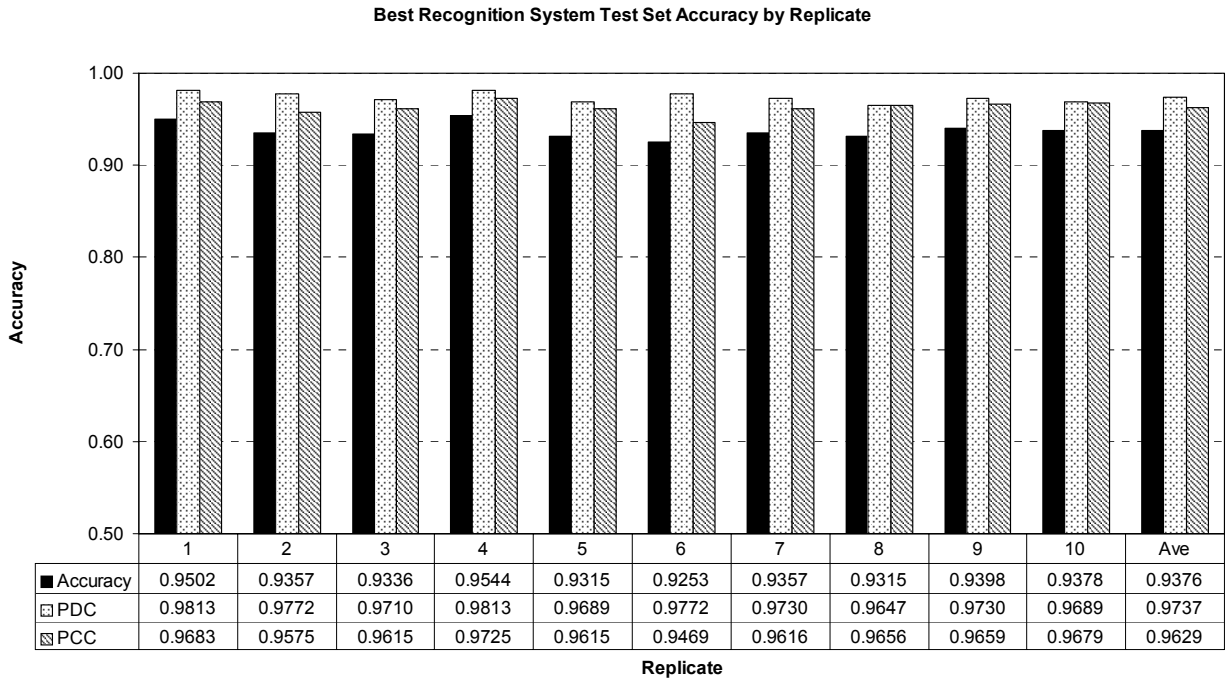
a prototype can receive a score from 0 to 6 points. We then total the points for all samples and divide by 6 times the number of samples; this produces a fitness score between 0 and 1. A score of 1 means that each prototype attracts the sample that was used to define it and the four samples that are + or - 6° and 12° away from the prototype in pose space. This suggests that the prototypes and their supporting features are generalizing well. This new performance measure allows us to use the maximum amount of training data available to form the classification system and still be able to measure the quality of the features.

4.2.3 MSTAR Experimental Results

4.2.3.1 Experiment 17-17-5

The HELPR system was used to synthesize a recognition system capable of multi-class target classification. Experiments were conducted using the publicly available MSTAR data set. In one series of experiments a recognition system was evolved using training samples spaced approximately 5° apart drawn from set of targets at a 17° depression angle spanning a full 360° pose space. The results of ten replicates of the recognition experiment are summarized in Figure 24. The typical probability of correct classification acheived by the evolve recognition systems is above 96% with a declaration rate of 97%. The gentle slope of the fitness curve shown in Figure 25 suggests that the system is indeed evolving solutions by using building block discovered in earlier generations. The trajectory of the test accuracy plot shown in Figure 26 closely follows the fitness curve. This provides evidence that the fitness measure is a reasonable predictor of generalization. Figure 27, plots the complexity of the best evolved recognition systems. The individuals in the initial population of recognition systems are constructed using 3 randomly generated transforms and 5 randomly positioned cap points for each transform. As the population evolves, the complexity of these recognition systems gradually climbs to an average of 11 transforms with a total of 63 cap points. Notice the complexity is not a monotonic function. In some generations, smaller solutions prove to be more effective.

The trend of the average fitness of the population (Figure 28) is very similar to the behavior of the best solution' fitness (Figure 25). This suggests that the population is converging to a stable solution. The population test accuracy is shown in Figure 29. These results are consistent



	1	2	3	4	5	6	7	8	9	10	Ave
Accuracy	0.9502	0.9357	0.9336	0.9544	0.9315	0.9253	0.9357	0.9315	0.9398	0.9378	0.9376
PDC	0.9813	0.9772	0.9710	0.9813	0.9689	0.9772	0.9730	0.9647	0.9730	0.9689	0.9737
PCC	0.9683	0.9575	0.9615	0.9725	0.9615	0.9469	0.9616	0.9656	0.9659	0.9679	0.9629

Figure 24. Final Accuracy of the Best System By Replicate.

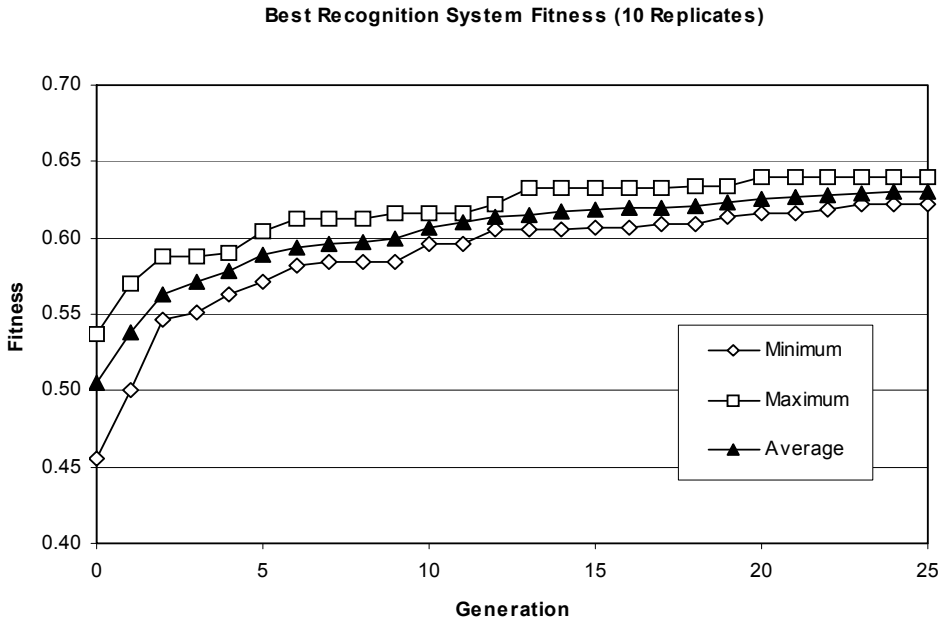


Figure 25. Average of the Best Recognition Systems' Fitness Score By Generation

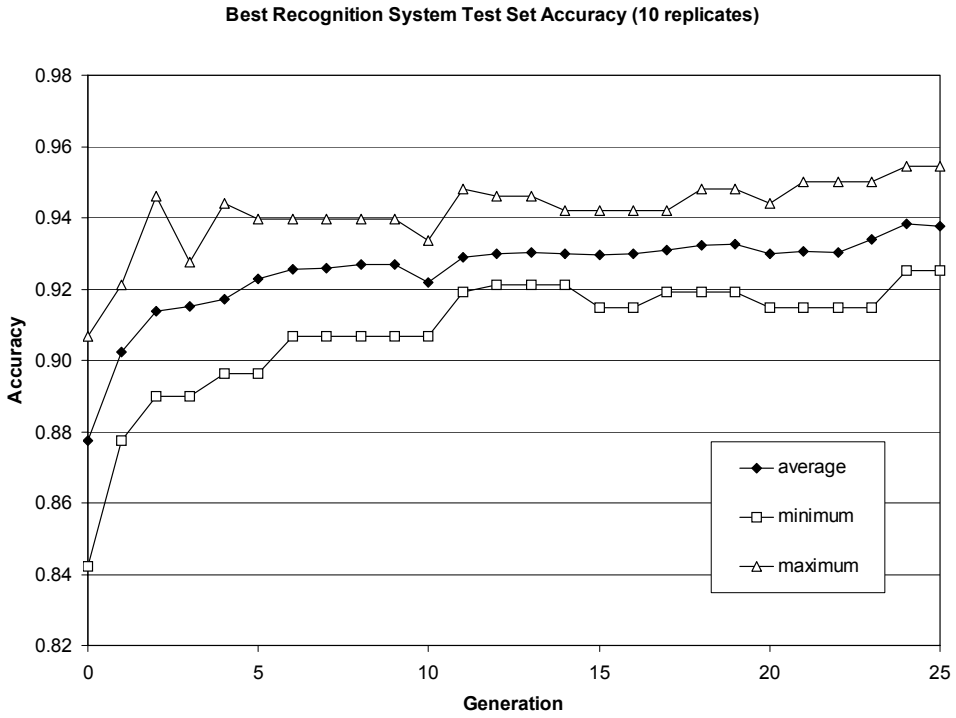


Figure 26. Average of the Best Recognition Systems' Test Accuracy By Generation.

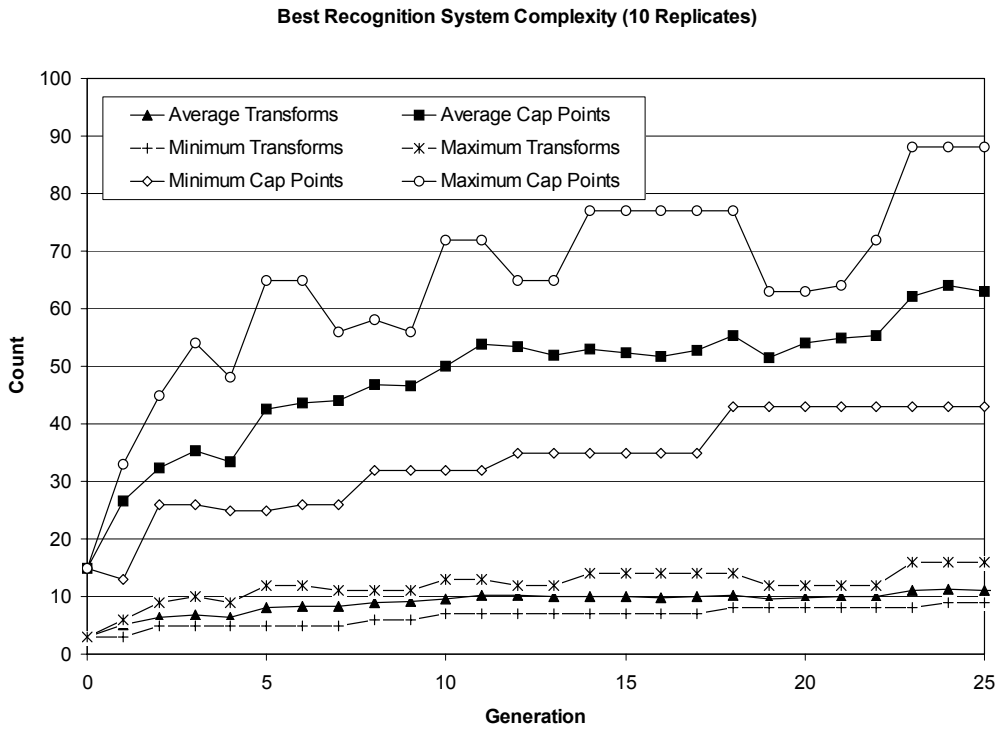


Figure 27. Average of the Best Recognition Systems' Complexity By Cycle

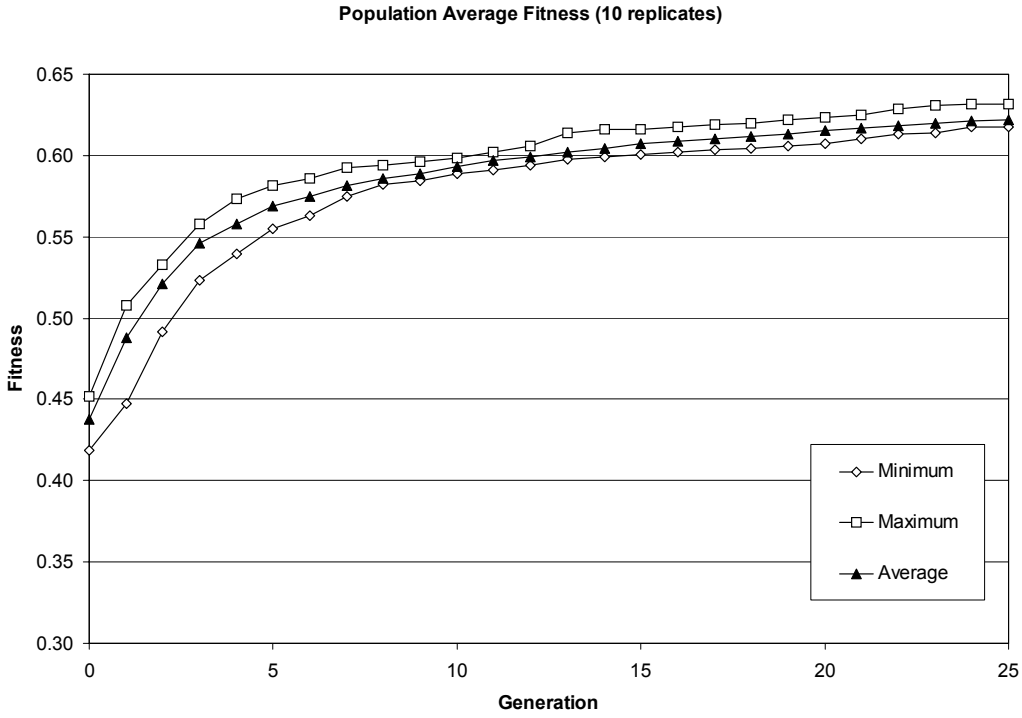


Figure 28. Average Population Fitness Across Replicates By Cycle

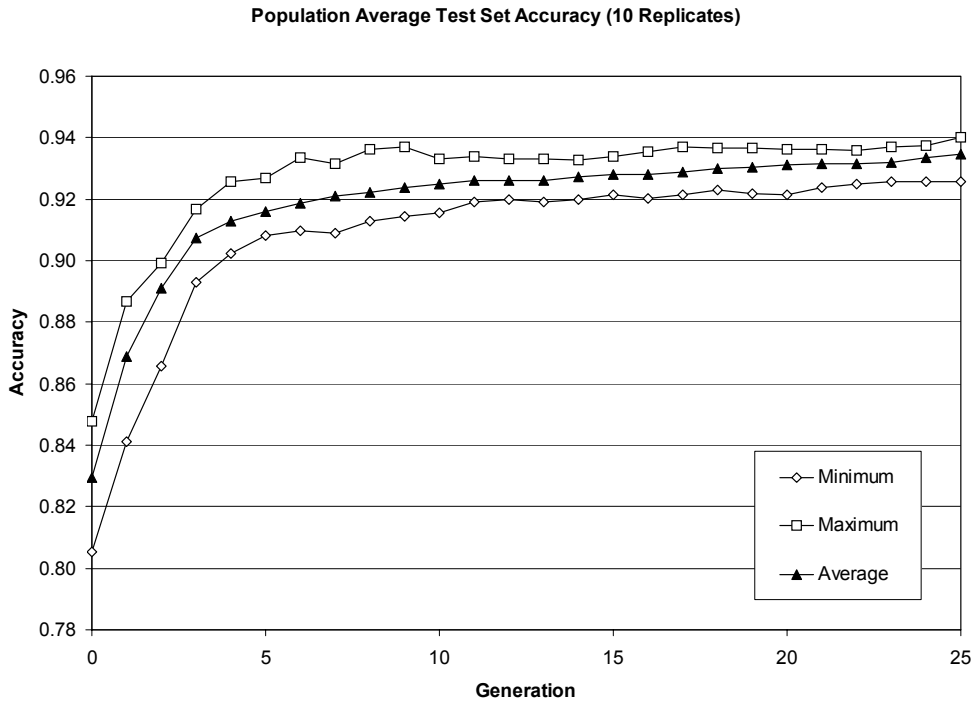


Figure 29. Population Average Test Accuracy Across Replicates By Cycle

with the observation that a recognition system's training fitness predicts its classification accuracy on the independent test data. The average number of transforms are displayed in Figure 30.

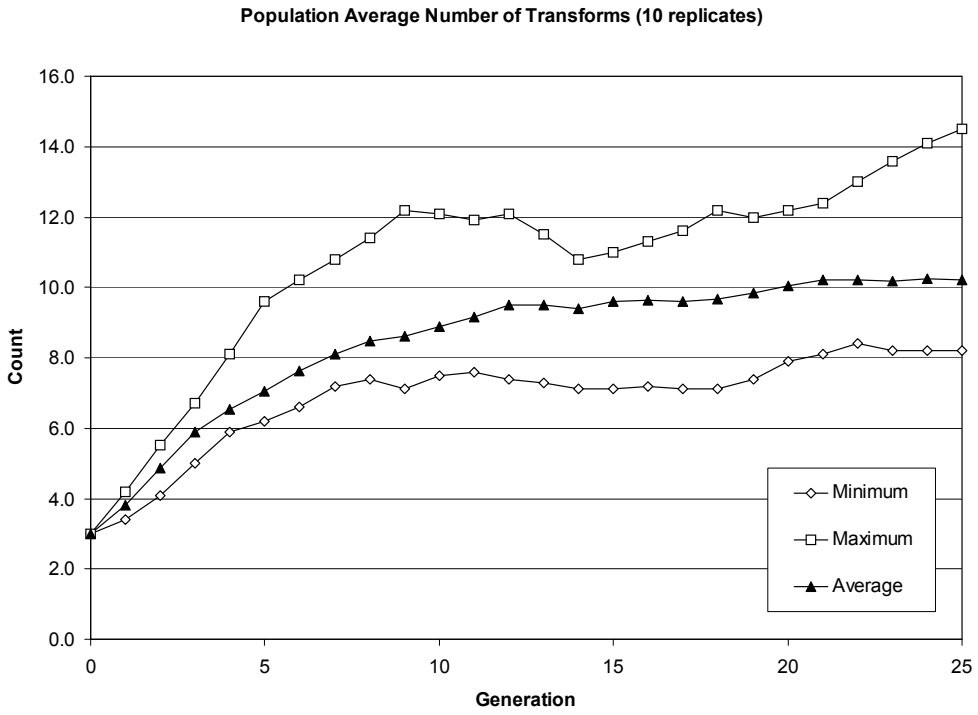


Figure 30. Number of Transforms Across Replicates By Cycle

The details of a single run are presented to help understand the behavior of the evolved recognition systems. These results correspond to the solution referred to as replicate 4 in Figure 25. This recognition system achieves a raw test set recognition accuracy of 95.44%. It declares a class label (PDC) for 98.13% of the targets and when it declares a class its probability of a correct declaration (PCC) is 97.25%. The confusion matrix for this recognition system is shown in Table 4. Notice that there are only a few misclassification and they are distributed fairly evenly among

Table 4: Confusion Matrix

	BMP	**BTR**	**T72**	**Other**
BMP	**148**	**3**	**3**	**7**
BTR	**2**	**158**	**0**	**1**
T72	**2**	**3**	**154**	**1**

the 3 classes. The relationship between the errors and the pose of the target are detailed in Figure

31. Most of the errors occur when the pose of the targets are near 0°, 90°, 180°, or 270° degree.

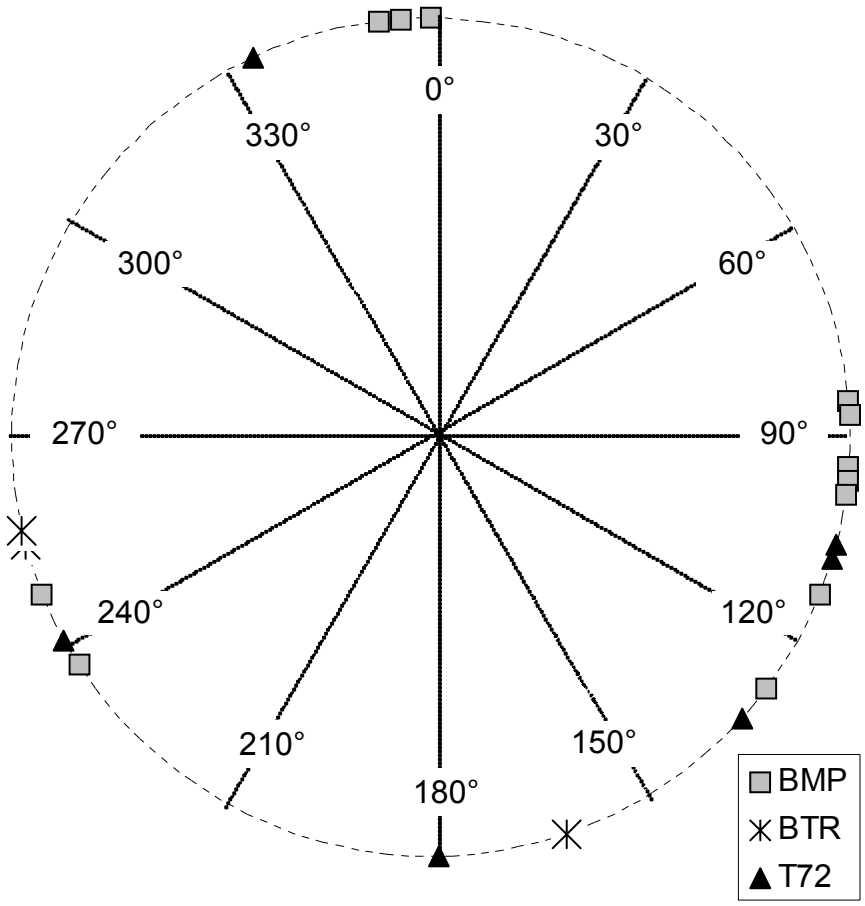


Figure 31. Errors in Replicate 4 By Pose Angle

Surprisingly the largest number of errors occur when the targets are viewed from the side. There is no obvious explanation for this behavior.

The details of the transformations and cap locations are shown in Figure 32. There are nine transforms used in this solution and there are between 3 and 8 caps applied to each transform. Most of the transforms are very compact consisting of only two morphological operators, but there are a few bloated expressions which is typical when GP is used to generate solutions. The output of 4 of the 9 transforms are shown in Figure 33. Each transform (T0, T1, T3, T8) is applied

```
T0: ( * ( Er ( Op ( Er ( Op ( Op ( I ) sqr 3 ) sqr 4 ) eVert 0 ) eVert 0 ) rHorz 2 ) ( Bd ( I ) eHorz 1 eHorz 2 ) )
T1: ( Cl ( Bo ( I ) eVert 0 eVert 2 ) cross 1 )
T2: ( Er ( Be ( I ) eVert 0 eVert 1 ) disk 4 )
T3: ( Di ( Cl ( I ) eVert 1 ) eHorz 2 )
T4: ( Er ( Er ( Be ( Er ( Er ( Bo ( I ) sqr 2 sqr 4 ) cross 0 ) cross 2 ) sqr 1 sqr 2 ) disk 1 ) cross 2 )
T5: ( Op ( Bo ( Cl ( Bo ( I ) sqr 0 sqr 4 ) rHorz 0 ) disk 2 disk 3 ) disk 3 )
T6: ( Be ( Be ( Be ( Be ( Be ( I ) cross 1 cross 2 ) eVert 1 eVert 2 ) cross 1 cross 2 ) sqr 3 sqr 4 ) eVert 0 eVert 2 )
T7: ( Di ( Di ( Bo ( Bo ( Di ( Bo ( Bo ( Di ( Bo ( I ) rHorz 1 rHorz 2 ) disk 3 ) rVert 0 rVert 1 ) eVert 0 eVert 1 ) rHor
    2 ) eHorz 1 eHorz 2 ) cross 1 cross 2 ) disk 1 ) eVert 2 )
T8: ( Bo ( I ) eHorz 0 eHorz 1 )

cap 0
    0 x=  45 y=  27 sz=  6 type=  0
    1 x=  19 y=  33 sz=  1 type=  0
    2 x=  17 y=  48 sz=  7 type=  0
    3 x=  25 y=  39 sz=  4 type=  0
    4 x=  20 y=  38 sz=  5 type=  0
    5 x=  42 y=  34 sz=  5 type=  0
    6 x=  35 y=  41 sz=  2 type=  0
cap 1
    0 x=  20 y=  46 sz=  5 type=  0
    1 x=  26 y=  33 sz=  6 type=  0
    2 x=  34 y=  34 sz=  4 type=  0
    3 x=  27 y=  46 sz=  1 type=  0
    4 x=  47 y=  42 sz=  1 type=  0
    5 x=  21 y=  46 sz=  3 type=  0
    6 x=  41 y=  29 sz=  3 type=  0
cap 2
    0 x=  39 y=  25 sz=  1 type=  0
    1 x=  38 y=  19 sz=  6 type=  0
    2 x=  19 y=  33 sz=  7 type=  0
    3 x=  30 y=  26 sz=  7 type=  0
    4 x=  31 y=  31 sz=  5 type=  0
    5 x=  43 y=  41 sz=  4 type=  0
cap 3
    0 x=  18 y=  45 sz=  3 type=  0
    1 x=  29 y=  16 sz=  4 type=  0
    2 x=  28 y=  24 sz=  4 type=  0
cap 4
    0 x=  22 y=  19 sz=  5 type=  0
    1 x=  20 y=  18 sz=  4 type=  0
cap 5
    0 x=  45 y=  20 sz=  2 type=  0
    1 x=  38 y=  18 sz=  4 type=  0
    2 x=  48 y=  41 sz=  2 type=  0
    3 x=  44 y=  23 sz=  7 type=  0
    4 x=  17 y=  38 sz=  1 type=  0
    5 x=  16 y=  41 sz=  3 type=  0
cap 6
    0 x=  33 y=  39 sz=  4 type=  0
    1 x=  28 y=  46 sz=  5 type=  0
    2 x=  27 y=  23 sz=  7 type=  0
    3 x=  34 y=  43 sz=  2 type=  0
    4 x=  18 y=  20 sz=  6 type=  0
    5 x=  23 y=  38 sz=  1 type=  0
    6 x=  24 y=  17 sz=  3 type=  0
cap 7
    0 x=  31 y=  48 sz=  7 type=  0
    1 x=  42 y=  34 sz=  3 type=  0
    2 x=  26 y=  43 sz=  4 type=  0
    3 x=  29 y=  31 sz=  1 type=  0
    4 x=  39 y=  40 sz=  5 type=  0
    5 x=  45 y=  16 sz=  3 type=  0
    6 x=  43 y=  17 sz=  1 type=  0
    7 x=  26 y=  32 sz=  3 type=  0
cap 8
    0 x=  29 y=  23 sz=  6 type=  0
    1 x=  24 y=  21 sz=  2 type=  0
    2 x=  31 y=  22 sz=  5 type=  0
    3 x=  17 y=  20 sz=  2 type=  0
    4 x=  39 y=  17 sz=  7 type=  0
```

Figure 32. Details of Transforms and Caps for Replicate 4.

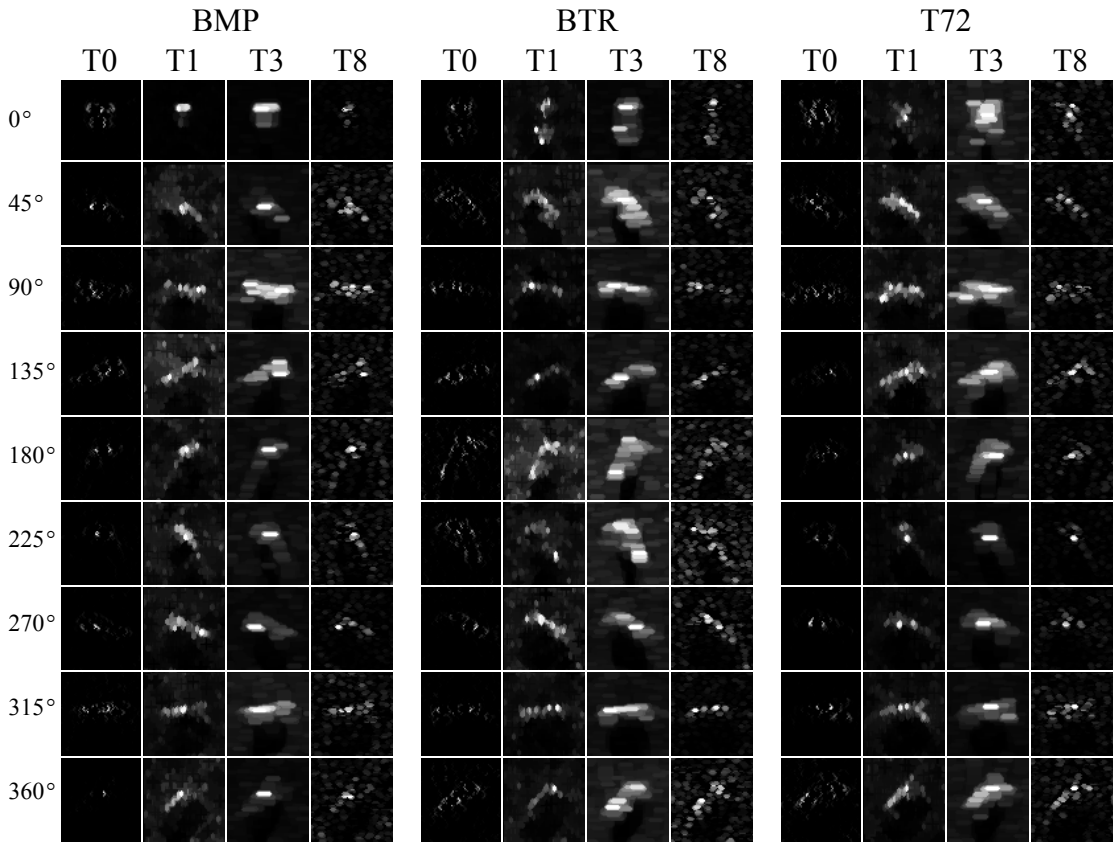


Figure 33. Output from 4 Sample Transforms

to the 3 targets (BMP, BTR,T72) and the effects are displayed by pose angle (rows). Clearly, the different transforms produce different patterns when applied to the different classes of targets. In Figure 34, the four caps are superimposed on the sample transforms. In some cases the caps overlap regions of high activity (bright areas) in the transformed images and in other cases they aligned with areas containing low activity (dark areas). This allows the caps to detect patterns of behavior. Also notice how caps can line up to detect extended area of activity. In some cases caps even overlap which has the effect of giving more weight to certain regions in the transformed images. The caps are pose sensitive so they can also be used to predict the pose angle of a target. In Figure 35, the difference between the predicted and actual pose angle for each target is displayed. Remarkably the same features that accurately predict class labels predict pose within +/-

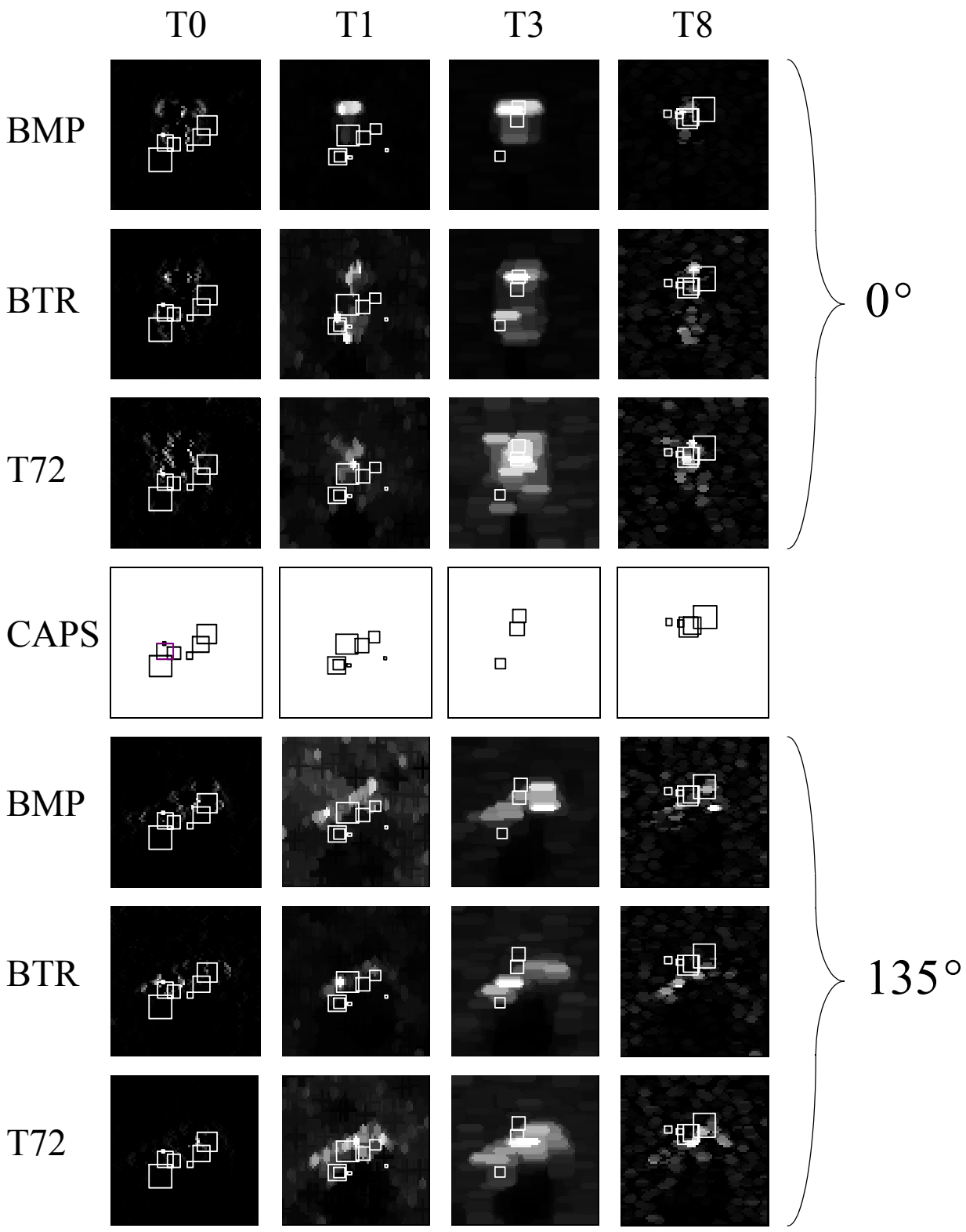


Figure 34. Caps Superimposed on Sample Transforms

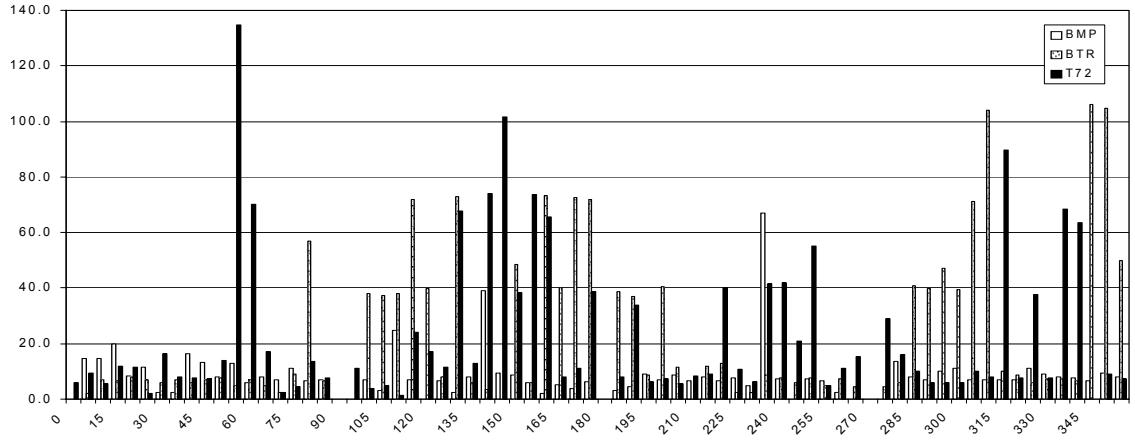


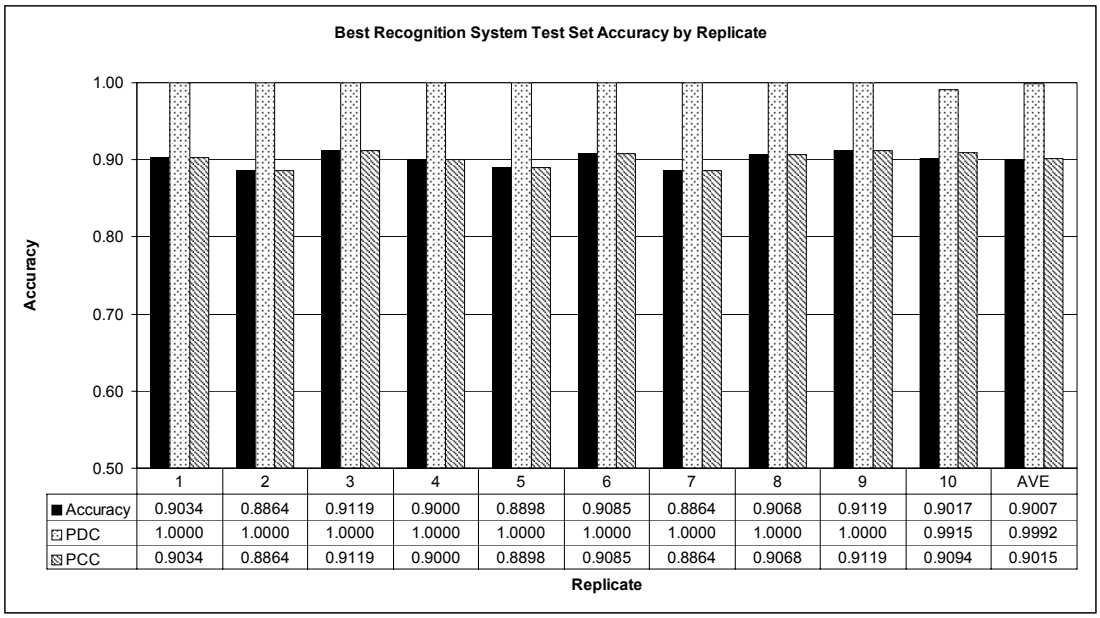
Figure 35. Predicted Pose Angle

2° degrees of accuracy. The worst errors tend to be +/- 180° where the front and the back of the target are not distinguished properly.

4.2.3.2 Experiment 17-17-10 and 17-15-5

To test the robustness of the HELPR generated solutions, two additional types of experiments were performed. In the first experiment described in the previous section training data was taken for 3 targets at a depression angle of 17° for training, 17° for testing and the training samples were spaced approximately 5° apart. In the 17-17-10 experiment both training and test samples are taken from targets at a depression angle of 17°, but the training data is spaced every 10°. In the 17-15-5 experiment training data is drawn from samples at a 17° depression angle while test data is taken from a data set with a 15° depression angle. The training data are space approximately 5° apart.

The accuracy of the 17-17-10 experiment is shown in Figure 36. Here we see a small degradation in the performance of the evolve recognition systems. The test set accuracy drops from approximately 96% to 90% accuracy when the spacing between training samples goes from 5° to 10°. The best recognition systems' fitness is very similar to the results obtained in the previous experiment, but the scores are slightly depressed (see Figure 37) compared to the result from the 17-17-10 experiment. The test accuracy of the best recognition system is also down (see Figure



	1	2	3	4	5	6	7	8	9	10	AVE
Accuracy	0.9034	0.8864	0.9119	0.9000	0.8898	0.9085	0.8864	0.9068	0.9119	0.9017	0.9007
PDC	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	0.9915	0.9992
PCC	0.9034	0.8864	0.9119	0.9000	0.8898	0.9085	0.8864	0.9068	0.9119	0.9094	0.9015

Figure 36. 17-17-10 Best Recognition System Accuracy

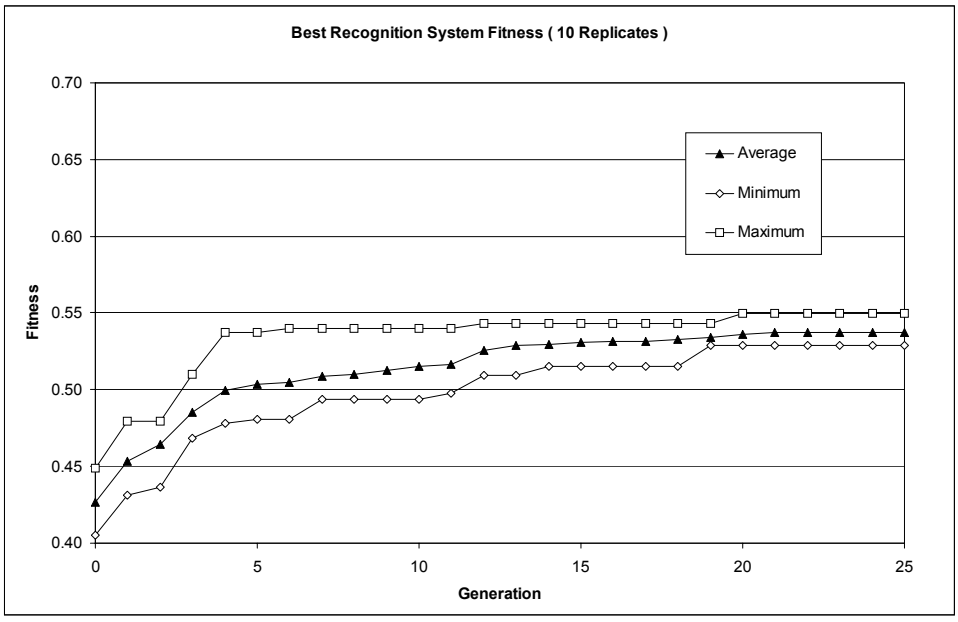


Figure 37. Average of the Best Recognition Systems' Fitness Score By Generation

38). Figure 39 show a surprising result. The complexity of the evolved systems is lower when the

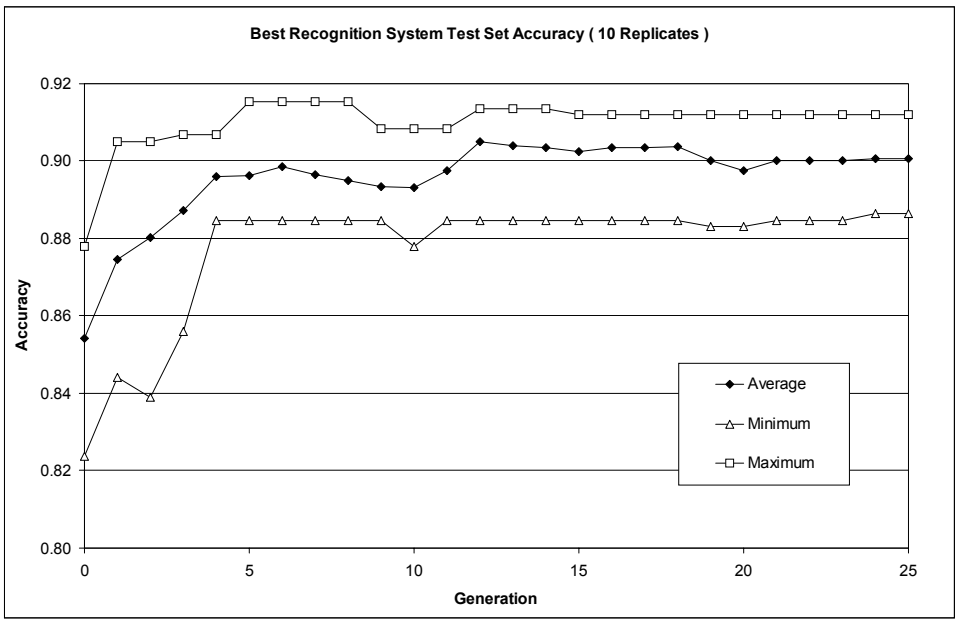


Figure 38. Average of the Best Recognition Systems' Test Accuracy By Generation

training data is spaced further apart. This may be a result of the reduced amount of information available during training. Essentially, the system has less information so it cannot create a highly customized system so it forms a less complex recognition system that in turn produces less accurate classifications.

The experiment evolves recognition systems using the training data taken from a data set sampled at a 17° depression angle and tests the evolve recognition systems using data taken from a set sampled at a 15° depression angle. These result are summarized in Figures 40, 41, 42 and 43. The typical test set accuracy for the 17-15-5 experiment is approximately 92%. This is slightly better than the 17-17-10 result. This is not surprising since the ability to interpolate behavior between targets with a 5° spacing between training instances should be easier than interpolating behavior between targets with a 10° spacing between training instances. In some sense the 17-15-5 experiment is slightly harder than the 17-17-5 experiment because there would be some additional error in the measurement of the depression angle. The results suggest that the evolved recognition systems are fairly robust. They can interpolate in a 5° x 5° window of azimuth and

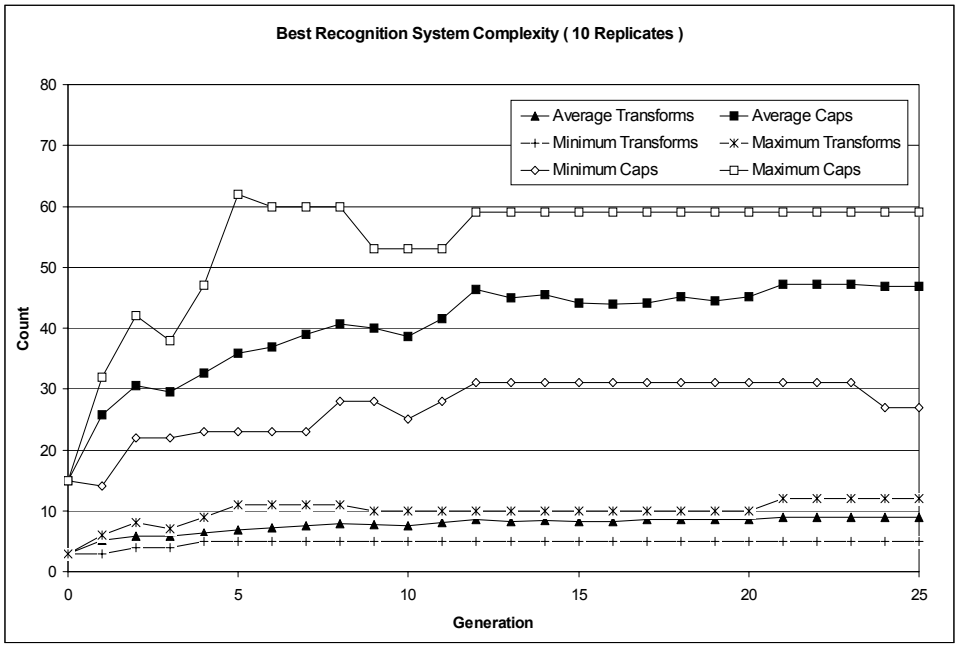
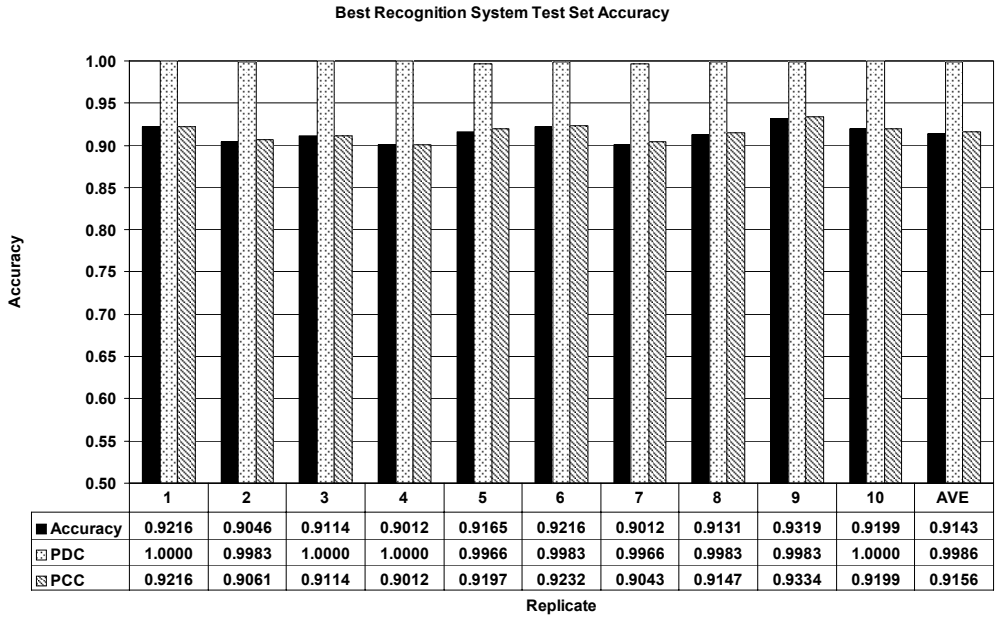


Figure 39. Average of the Best Recognition Systems' Complexity By Cycle



	1	2	3	4	5	6	7	8	9	10	AVE
Accuracy	0.9216	0.9046	0.9114	0.9012	0.9165	0.9216	0.9012	0.9131	0.9319	0.9199	0.9143
PDC	1.0000	0.9983	1.0000	1.0000	0.9966	0.9983	0.9966	0.9983	0.9983	1.0000	0.9986
PCC	0.9216	0.9061	0.9114	0.9012	0.9197	0.9232	0.9043	0.9147	0.9334	0.9199	0.9156

Figure 40. 17-15-5 Best Recognition System Accuracy

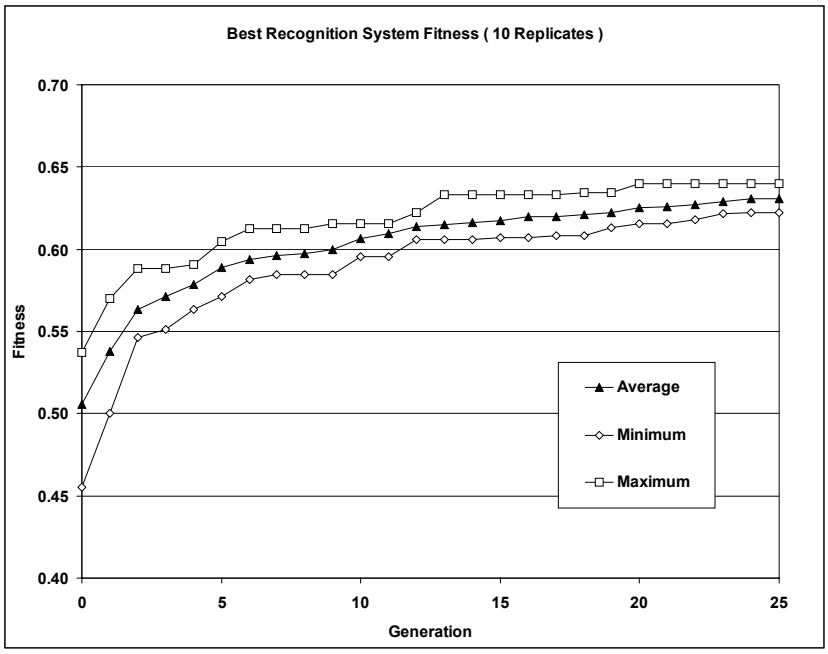


Figure 41. Average of the Best Recognition Systems' Fitness Score By Generation

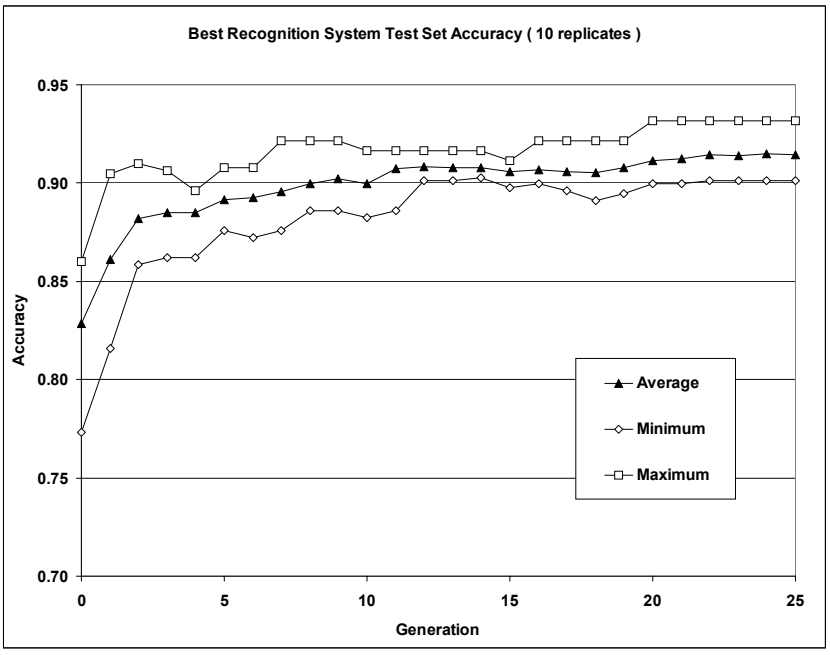


Figure 42. Average of the Best Recognition Systems' Test Accuracy By Generation

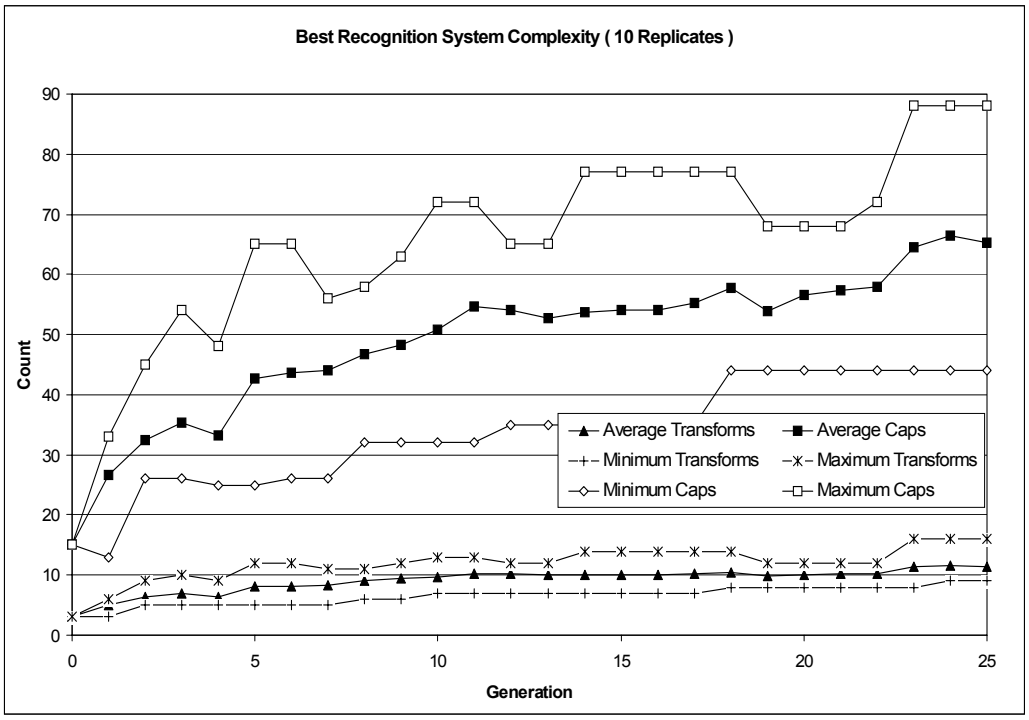


Figure 43. Average of the Best Recognition Systems' Complexity By Cycle

elevation with fairly good accuracy. Notice the 17-15-5 experiment shows an increase in the complexity of the evolve recognition systems (see Figure 43). Again this is not surprising given that the system has the information it needs to develop an accurate recognition system, but it must solve a slightly harder problem.

4.2.4 Discussion of the SAR Results

The evolved recognition system achieved accuracies above 95% with high declaration rate. Furthermore the results show that the solution often require less than 10 transformation and 5 caps per transform. This means the system evolves 72 prototypes containing less than 50 (10 transforms * 5 caps) scalar measurements per target that accurately classify targets at a full range of poses 0-360 degrees with a variation in elevations of +/- 2.5°. Additional results suggest that the number of prototypes could be reduced significantly since the amount of variation in the target is not uniform across the pose space. Although reducing the number of prototypes would also reduce the accuracy of pose prediction. If classification accuracy is the primary concern, then a

simpler classification system can be formed using sensitivity analysis to remove redundant prototypes from the evolved target recognition system.

4.3 E3D Application

The HELPR system was also used to evolve pattern recognition systems for the E3D Challenge data set. The challenge data set is provided in the form of point clouds. These point clouds were voxelized to form volume images. The front end module of HELPR was again modified to perform 3D morphological processing and the neural network classifier system was replaced with a KNN classifier. The tank training problem was used as an experimental testbed. This problem requires discriminating among two groups composed of 10 different tanks and 26 confusers respectively. The system was trained using 6 tanks at one pose angle. The results demonstrate that by varying the radius of attraction of the KNN classifier, recognition accuracies approaching 90% can be achieve training the system using only a single pose.

Classification of 3D objects is becoming an increasingly important research area due to cheap and innovative sensor technology. Shadows, noise, viewing direction, and distance from the sensor all directly affect the quality and amount of surface information provided by the sensor. The recognition approach described in this paper converts surface information, a set of (x,y,z) points, into a discrete 3D binary image. This conversion step processes the surface points using a fuzzy technique to mitigate the effects of noise and minor distortions. These images are then processed by sequences of one or two randomly selected morphological operators. Each of the sequences' output is then fed into a simple transducer to obtain a set of scalar feature values. The feature values are classified using a K nearest neighbor (KNN) classifier that is trained using a sparse number of training samples. Experiments were conducted using the Air Force Research Laboratory's E3D data and experimental protocol. Experimental results for the tank classification problem using 10 tanks and 26 confusers are presented. The results show the combination of morphological processing and KNN classifier produced consistently good performance under variations in noise, viewing angle, or distance.

4.3.1 E3D Overview

How one formulates a problem often influences the results obtained with a designed pattern recognition system. Currently, the Air Force Research Laboratory (AFRL) HELPR (Hybrid Evolutionary Learning for Pattern Recognition) research effort is using 2D mathematical morphology and evolutionary computing to design pattern recognition systems. The E3D data distributed by AFRL [2] is a cloud of data points specified by a list of (x, y, z) coordinates. A transition of the HELPR approach to the E3D data base initiated a decision to use a 3D voxel structure which is a natural extension of a 2D image pixel array. Each voxel is a square unit of volume specified by three indices. This paper describes an initial investigation into pattern recognition and 3D mathematical morphology applied to a standard E3D data and problem set. Examples of the cloud data are presented along with a discussion of how we transform the cloud data into a voxel representation. Some information is lost with this quantization of real coordinates. A process called fuzzy voxelization addresses this quantization effect. Also presented are preliminary experimental results with mathematical morphology and the voxel representation. The results demonstrate the potential of the voxelized representation for future 3D ATR research efforts.

4.3.1.1 Three dimensional Challenge data

LADAR sensors image their environment by emitting a laser beam that hits an object and returns to the sensor. By measuring the beam's time of flight, the object's distance away from the surface can be determined. By scanning repeatedly in small increments horizontally and vertically, the entire object is sensed in small surface patches and an accurate representation of the surface's geometry can be obtained. This relatively new type of data offers the possibility of new target recognition algorithms. Toward this goal, AFRL has established a "Challenge Problem". This problem is defined as a series of experiments on a database of 80 military vehicles. The database of targets includes surface information of all vehicles under 5 different viewing angles (viewing angle = 0°, ±15°, and ±30°; nominal elevation angle = 45°) and 3 different ranges (100m, 500m, and 1000m). In total, there are 1200 (i.e., 80·5·3) data files. Software in the distribution is provided to create noisy versions of the surfaces. The series of experiments defined in

the Challenge problem are discussed in section 4.3.4. Figure 44 shows CAD models for 6 targets and Figure 45 shows examples of point clouds (set of x-y-z coordinates) provided by the sensor. The leftmost figure in Figure 45 shows the result when the sensor is oriented -30° from head-on while the second image is generated when the sensor is +30° from head-on. Notice that the point clouds have shadows created by the surfaces that obscure other parts of the target. For example, the tank's barrel causes a shadow that appears as the empty region located at the lower-front portion of the tank. Also, by varying the sensor's viewpoint, portions of the vehicle's surface disappear, while others portions become visible. For example, the JSU's right wheel tracks are visible at -30° but almost completely obscured at +30°. Even though a portion of the visible surface remains constant from different vantage points, the differences are still significant and present a difficult recognition problem.

Figure 44. Vehicles from Challenge Data.

The images are not shown to scale.

Figure 46 illustrates the effects of additive noise. The leftmost image shows a Hummer without

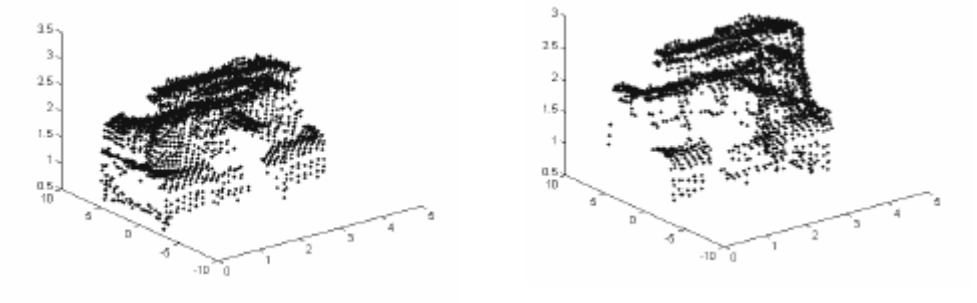


Figure 45. JSU 122 point clouds obtained from different viewpoints.

The left and right figures show the noiseless surface points obtained when viewing the JSU 122 from -30° and +30°, respectively.

noise and the rightmost image shows the same vehicle with 72mm of noise, where the 72mm rep-

resents the standard deviation of a Gaussian distribution from which the noise is sampled. Each of the original points is moved a random amount, forward or backward, along the sensor's line of sight. This randomness provides less precise images, as illustrated in the righthand part of Figure 46.

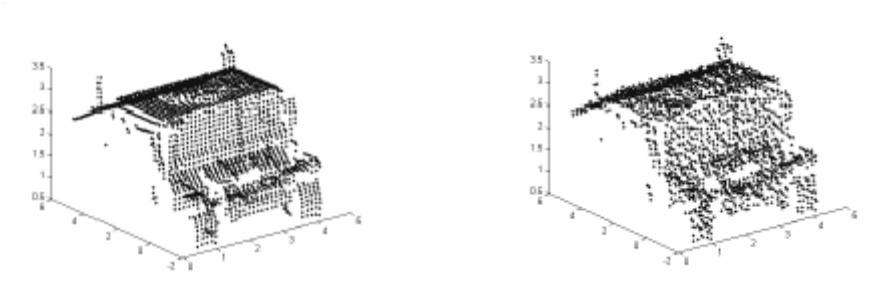


Figure 46. Hummer with and without noise.

These point clouds show the hummer with 0mm and 72mm of noise, with the sensor oriented at 0° and located 1000m from the vehicle.

The third operating condition is distance of the sensor from the object. Increasing the sensor's distance yields fewer points on target, making the density of points inversely proportional to the sensor's distance from the vehicle. For example, one typical vehicle in the set imaged at 100m, 500m, and 1000m had 215000, 8000, and 2000 points on target, respectively. The point density has implications on recognition algorithms as they should function properly with sparse or dense data.

4.3.2 Morphological Processing

Mathematical morphology [14, 51] is a method of transforming an image according to it geometric properties. The basic morphological transformations produce results that are based on criteria that can be interpreted by humans such as shape, size, orientation, texture, etc. Mathematical morphology is versatile and has been applied to one-dimensional signals, grayscale images, color images, and graphs. The morphological definitions for binary image are based on crisp

logic, where the most common operation, erosion, is essentially a perfect template match. Extensions such as soft morphology [35] have been developed that are tolerant of noise and distortions in the image. This section describes the crisp morphological operators that are relevant to this work; readers who are interested in a more detailed treatment are referred to [51].

4.3.2.1 Voxelizing process

In order to utilize digital image processing techniques, it is convenient to place the set of real-valued coordinates into a discrete three-dimensional binary grid. This conversion, referred to as *voxelization*, is performed by superimposing a large uniform 3D grid over the spatially distributed points. If one or more points fall within a pixel, that pixel is set to true. One problem with this process is its dependency on exactly where the grid is positioned. For example a cluster of points may fall entirely within a single cell, or distributed into two or more cells if the grid is shifted slightly. To lessen the effects of this anomaly, the voxelization process is made less stringent by sub-dividing each pixel into a 3x3x3 region. Points that fall in one of the border subregions are also credited with residing in one or more neighboring pixels. Figure 47 illustrates this fuzzy voxelization process in two dimensions. For example, points that fall in the upper-left corner are considered to also reside in the three other neighboring pixels. The process illustrated in Figure 49 is easily extended to three dimensions. In this work, the elements of the grid (i.e., the pixels) represent a 0.4m x 0.4m x 0.4m region.

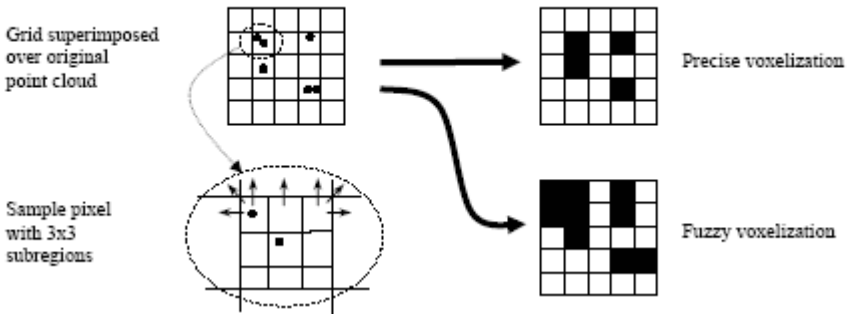


Figure 47. Voxelization process.

4.3.2.2 Morphological operations

Erosion is the fundamental operation in mathematical morphology (MM). It is parameterized by a special image called a structuring element (SE), which serves as a geometric probe. SEs are typically based on standard geometric shapes (e.g., disks, rectangles, and lines in 2D morphology). Erosion identifies the pixels in the image where the SE can be placed and still fit entirely within the image's "on" pixels. This process is essentially a template match. Eq. 1 defines erosion, where Is is the image translated spatially by the vector s. Erosion generally shrinks the image and completely removes particles that are smaller than the SE. Dilation (Eq. 2) is a closely related operator that enlarges the image, effectively closing small gaps. Dilation can also be viewed as eroding the background.

$$I \Theta S = \bigcap_{s \in S} I_{-s} \qquad \text{Erosion} \qquad \text{Eq. 4.3-1}$$

$$I \oplus S = \bigcup_{s \in S} I_{s} \qquad \text{Dilation} \qquad \text{Eq. 4.3-2}$$

Opening and closing (Eqs. 3 and 4) are two composite operators based on erosion and dilation. Opening is defined as erosion followed by dilation; whereas; closing is the reverse of this process. Unlike erosion and dilation, which substantially change the size of the image, opening and closing generally preserve the original image but remove particles (opening) or close up gaps (closing).

$$I \circ S = I \Theta S \oplus S \qquad \text{Opening} \qquad \text{Eq. 4.3-3}$$

$$I \circ S = I \oplus S \Theta S \qquad \text{Closing} \qquad \text{Eq. 4.3-4}$$

Each of the operators discussed above has an additional form called a band operator. A band operation is defined to be the difference between results produced by applying an operator using a SE shape at two different scales (Eqs. 5 and 6). These equations use Sa,b to denote a particular SE shape at sizes a and b (e.g., Cube1,3). The band operators are able to isolate surfaces of a particular range of sizes and can accentuate size differences.

$$I^{\circ}S_{a,b} = I^{\circ}S_a - I^{\circ}S_b \qquad \text{Band Opening} \qquad \text{Eq. 4.3-5}$$

$$I \bullet S_{a,b} = I \bullet S_b - I \bullet S_a \qquad \text{Band Closing} \qquad \text{Eq. 4.3-6}$$

A single morphological operator can remove irrelevant information and/or highlight salient features. When applied with a properly selected SE, the operator can be used as feature extractors for pattern recognition. Figure 48, for example, shows the result of opening a JSU (tank) and GMC CCKW 353 (non tank) with a 3.6m x 0.4m SE parallel to the ground and oriented across the width of the vehicles. This operation removes all surfaces except wide horizontal surfaces. Notice that this operation retains much of the JSU, while removing most of the GMC CCKW 353. By summing the number of on pixels in each output image, a scalar feature can be obtained that may help discriminate tanks from non-tanks. Features such as these, which emphasize inter-class differences, can be valuable to the classification process.

4.3.2.3 Morphological sequences

As discussed in the previous section, a single morphological operation can serve as a discriminating feature. Our earlier work as shown that additional discriminating capability can be obtained by allowing multiple operations to be applied sequentially and/or combined using set operations [48, 60]. By allowing arbitrary combinations, the flexibility of the representation is increased. The main problem, however, is identifying the useful sequences out of a very large

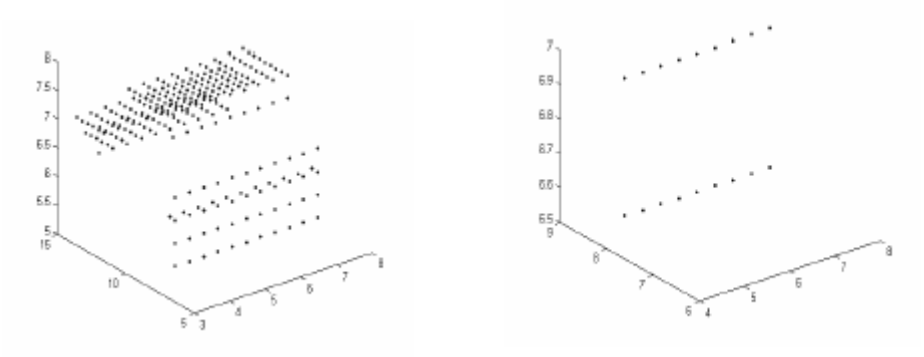


Figure 48. Morphological opening.

The left and right images correspond to opening the JSU 122 and GMC CCKW 353 vehicles.

number of possibilities. One research focus has been to learn collections of sequences that can be used for pattern recognition. By representing these sequences as expression trees, genetic programming [36] can be used to search for useful expressions. The work described here is the first step in this direction. Its purpose is to establish whether morphological sequences are as effective in 3D as observed in 1D and 2D recognition problems.

4.3.3 Classification System

The Challenge problem is a classification problem where the system trains on a set of targets and is expected to be able to identify other instances of targets. In the tank classification problem, for example, training is done on several types of tanks and is then tested with: the original tanks, unseen tanks, and unseen non-tanks (confusers). This type of training uses only targets and sharply contrasts many forms of learning that train on both positive and negative instances. The elements of training are:

I_i	$1 \le i \le I$	Set of I training targets
M_j	$1 \le j \le J$	Set of J morphological sequences

Figure 49 illustrates the design of the pattern recognition system used for the E3D application. J random morphological sequences process the input image to produce J transformed images. Each image is fed into the Σ, which sums the number of on pixels in the transformed image. The set of scalar values, the feature vector, is fed into a K nearest neighbor (KNN) classifier [16]. The KNN has a set of stored prototypes obtained using the training data. The unknown feature vector is then compared to the set of prototypes to find the closest one. If the closest one is closer than a specified maximum distance, it is declared a target; otherwise, a class is not declared.

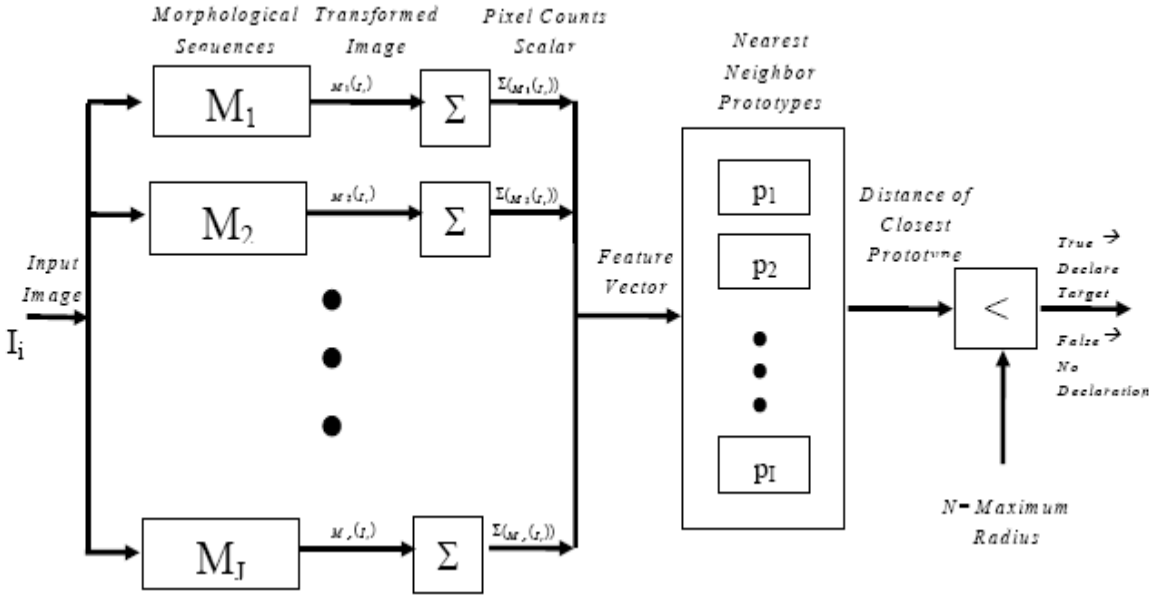


Figure 49. Pattern classification system.

The prototypes defined in the KNN classifier are directly obtained using the training data:

$$\vec{p}_i = \langle p_{i,1}, p_{i,2}, \ldots, p_{i,J} \rangle = \langle \sum(M_1(I_i)), \sum(M_{2(1)}(I_i)), \ldots, \sum(M_J(I_i)) \rangle \qquad (1 \le i \le I)$$

Ideal morphological sequences will transform target vehicles in a similar manner (e.g., leave a large residual), while simultaneously processing non-target vehicles in a complementary manner

(e.g., leave a small residual). By accentuating intra-class similarities and inter-class dissimilarities, a relatively simple classifier can be used to determine if the vehicle is a target or not.

There are a number of ways to measure the distance between feature vectors. In this work, the distance between a prototype and an unknown is defined as 1-cos(Θ) which is proportional to the angle formed by the prototype and unknown feature vector. Of course, cos(Θ) can be computed by taking the normalized dot product of the two vectors. Euclidean distance was also examined as a distance measure but was found to give inconsistent performance. Although the reason for its poor performance has not been definitively determined, it is conjectured that the un-normalized, widely varying, feature vectors allowed certain features to dominate the distance computation. The angle-based measure operates more gracefully with un-normalized vectors.

4.3.4 Experimental Results

4.3.4.1 Experimental protocol

The draft of Challenge problem [2] defines 4 main experiments – tank classification, vehicles with missiles/rockets, vehicles with track drives, and vehicles without guns. For each problem, 45 sub-experiments are defined, where a typical experiment requires training on a set of targets at viewing angle of 0°, 0mm of noise, and the target positioned 500m away from sensor. Testing is then performed using targets and confusers under a different condition (e.g., ±15°, 48mm noise, and 1000m distance).

Table 5: Vehicles used for tank classification problem.

Tanks	**Confusers**			
**** Used for training**	**Easy**		**Moderate**	**Difficult**
BT5**	BRDM-1	BRDM-1 w/ Snapper	BMP-1	FlackPanzer Gepard
Challenger 1**	BRDM-3	BTR-70 APC	M110-A2	Jagdpanzer
CV 35**	Dump truck	EMT	M2A2 Bradley	M109 A2
JSU 122**	Ford GPA Jeep	SA9 Gaskin		Scorpion

Table 5: Vehicles used for tank classification problem.

KV 1B	BAZ67B	GMC CCKW 353		Type 38
KVII**	M977 HEMTT	M109 A6 Paladin		
Leclerc**	Hummer	JTTRAI		
Leopard 1	Leyland	Magnum		
M4 A3 W Sherman	Mercedes L3000	Army cargo truck		
Panzer III				

Table 5 lists the vehicles used in these experiments. Ten different types of tanks are listed, with six being used for training. The confusers are listed under three subcategories, easy (18), moderate (3), and difficult (5), which is AFRL's subjective rating of how similar they are to the tanks. Certain vehicles from the distribution disks were omitted from these experiments when it was observed that they were not positioned in the scene as the vehicles listed in Table 5 were. Some of these other vehicles were rotated on the ground by ±45° or ±90°. While these rotated versions could have been used as confusers in these experiments, they were omitted completely. Other researchers using this data should also be aware that the point cloud for the EMT at the 100m and 0° pose is delivered in a byte order that is reversed relative to the other vehicles (e.g., big-endian/little-endian issues). We found only one example of this anomaly in the vehicles shown in Table 5.

The experiments described in this paper use a subset of the 45 experiments defined in the Challenge problem. In all experiments, the training data consisted of 6 training tanks at 500m, 0mm noise, and 0° viewing angle. Testing was then performed on all of the vehicles listed in Tab. 1, while varying one of the operating conditions. Table 6 shows the different conditions used for testing and the corresponding Challenge experiment numbers. In total 8 unique experiments are considered in this paper. It should be noted, however, many other experiments were conducted by simultaneously varying noise, distance, and pose; the observations made were substantially equivalent to those observed in the experiments presented here.

Table 6: Test Experiments

Noise	Distance	Pose	Challenge Experiment Numbers
0mm	500m	0°, ±15°, ±30°	2, 5, 8
0mm	500m, 1000m	0°	2, 3
0mm, 24mm, 48mm, 72mm	500m	0°	2, 11, 20, 29

This tank classification problem is non-trivial: 1) The amount of training data is very small, consisting of 6 targets under a single condition (500m distance, 0mm noise, and 0° vantage point), yet is asked to capture the general concept of tank. 2) There are significant intra-class differences among tanks, as can be seen by the considerable structural differences of the three tanks shown in Figure 3) Testing is performed using noise, distance changes, or viewpoint changes that yield point clouds that significantly differ from the point clouds seen during training.

The primary measurements used to evaluate a recognition system are target detection and false alarm rate. Obviously, it is desirable to have high detection rates and low false alarm rates. It is usually difficult to state a single pair of accuracies, however, since most systems have one or more configurable parameters that increase/decrease the declaration rate. For example, the nearest neighbor classifier used in these experiments has a maximum radius of attraction associated with each prototype. If the distance of an unknown feature vector to the closest prototype exceeds this radius, no class is declared. That is, the sample is not close enough to a prototype to be confidently declared a target. Small maximum radii require an unknown to be very similar to one of the prototypes. As the radii increase, the prototypes capture more and more of feature space. Although a large radius will allow the declaration of more tanks, it is inevitable that some confusers (i.e., non-tanks) will be called tanks. Thus, the radius can be modulated to aggressively declare tanks, but with the risk of higher false alarm rates. Figure 50 illustrates the effects of varying the maximum radius of attraction for a sample classification system. For this system using a maximum radius of 0, an unknown feature vector must be identical to one of the prototypes in order to be declared a target. When testing conditions differ from training conditions, as was the case in

this figure, a maximum radius of 0 will generally yield no declarations at all. As the maximum radius increases, the number of declared tanks increases, as does the number of false alarms. By adjusting the maximum radius, a range of behaviors can be produced. The rightmost plot in Figure 50 displays this information in plot called a receiver operator characteristic (ROC) curve. This plot eliminates the KNN maximum radius from consideration and simply displays the tradeoffs between the probabilities of detection and false alarm. This particular ROC curve shows that a 90% declaration rate can be obtained with a false alarm rate of about 20%. The ideal ROC would be a step function showing 100% detection rate with 0% false alarms, although such curves are difficult to attain in practice.

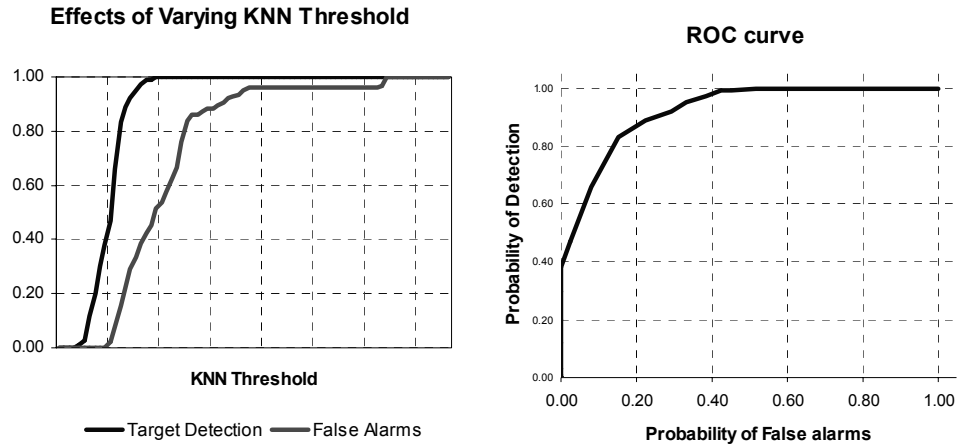


Figure 50. Effects of KNN max. radius and probabilities of detection / false alarm.

4.3.4.2 Recognition results

The system defined in Figure 49 was populated with J=50 random morphological sequences. Each sequence consists of 1 or 2 operators selected randomly from the set of operations including: opening, closing, band-opening, band-closing. Four types of SEs were used: a cube and 3 bar shapes. One bar is parallel to x-axis, one is parallel to the y-axis and the third bar is parallel to the z-axis. The sizes of the SEs ranged from 1.2m to 4.4m, in 0.8m increments. During

the training phase, the sequences' only screening criteria was that they had to produce some variation on the training targets. This simple test removed trivial sequences such as those that eliminate all pixels from all images or fill the images up completely. Such sequences are useless and therefore are prevented from entering the system.

All training was performed on the six training targets at 0mm noise, 0° viewing angle, and 500m distance between sensor and vehicle. Thus, six prototypes were used in the KNN. As prescribed in the Challenge problem, all tanks and confusers were then classified by the system at a particular operating condition. By systematically altering one of the operating parameters it is possible to analyze the effects that parameter has on the recognition system. For example, the upper leftmost plot in Figure 51 shows the system's ROC curves under different viewing angles.

As expected, the plots show the ROC curve shifted down and to the right slightly with changes to viewing angle, indicating decreasing performance. This is not surprising since even a 15° change in viewing angle significantly changes the point cloud. The upper rightmost plot in Figure 51 shows four ROC curves illustrating the effects of noise. The four curves are generally equivalent. This is expected, as even the largest amount of noise (0.072m) is much less than the scale of the voxelization (0.4m). As a result, the initial fuzzy voxelization process effectively eliminates the effects of noise at these levels. The bottom-most plot of Figure 51 shows that effects of changing the sensor's distance from the vehicle. Here too, the curves generally exhibit equivalent behavior. The reason for this is two-fold. First, even at 1000m, the points are relatively dense on the vehicle's surface. Second, the voxelization process is relatively unaffected by increasing/decreasing the density of points, as long as the resolution of points on target is finer than 0.4m.

In general, the ROC curves indicate mathematical morphology is a promising way to process the 3D data. With randomly generated sequences, and simple screening criteria, reasonable ROC curves are produced. For example, an 80% declaration rate can be obtained with a 25% false alarm rate. It is expected that considerable improvement can be made with more intelligent meth-

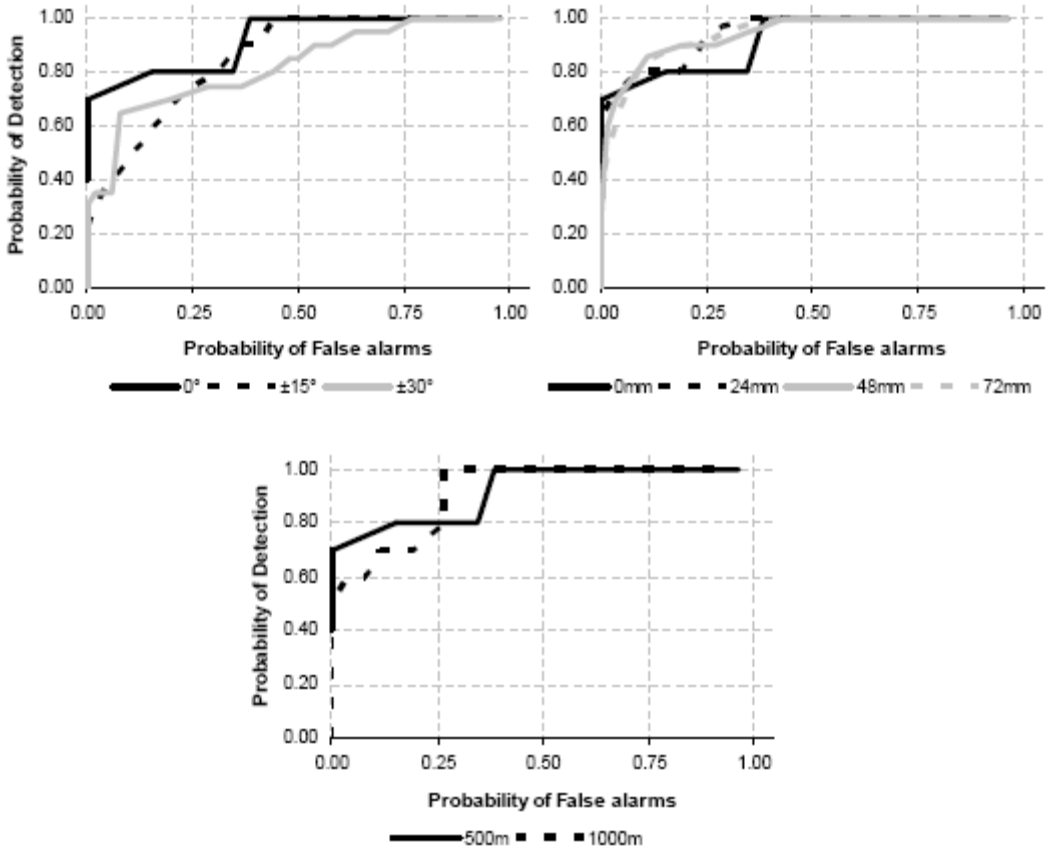


Figure 51. Effects of varying operating conditions.

ods for generating sequences and better techniques for locating cooperative subsets of sequences.

Although not shown, ROC curves were also generated for the specific subclasses of confusers (i.e., easy, moderate, and difficult). Generally there was no observable difference between the system's performance on the different subclasses of confusers, suggesting the system's concept of similarity differs from human judgment.

4.3.5 Learning Experiment

For the learning experiment, results for a data set composed of 5 military targets are generated. In this experiment, 100 feature detectors (templates) are generated and applied to a training

set composed of 8 samples of each target (5 targets x 8 poses). The eight samples represent pose variation in the range of 0º to 360º with increments of 45º. The test set is composed of the same targets generated at poses in the range of 0º to 360º in increments of 15º (items duplicated between the training and test set are removed from the test set). This provides 40 test samples of each target or a total of 160 test samples. This is a fairly simple experiment that quickly yields 100% accuracy for the training set (40 out of 40 correct) and 99.4% accuracy on the test accuracy (159 out of 160 correct).

4.3.5.1 Morphological Feature Extraction

The feature detector subsystem of HELPR involves three steps: feature extraction, feature selection, and feature integration. For point cloud data, we are only using the first two steps. Morphological features are extracted using hit-and-miss templates that are used to probe the voxel surface representation of a point cloud. Templates are stochastically generated. Each template is composed of a mixture of foreground points that are required to fit into voxels on the surface of the object that while background points are required to fit into voxels in the image that are turned-off. There is a strong bias toward the selection of foreground points so the focus is on finding patterns that represent the surface of the object. Background points are useful for identifying areas that must be open. For example, to recognize whether a dump truck is loaded or not requires a set of features that are sensitive to the empty area in the back of the truck. Templates are generated within a rectangular volume that encompasses approximately one-third of a typical target. This forces features to be somewhat localized to a specific region of the target. As a result individual features are less sensitive to occlusions and noise. There is some rational for allowing templates of many different sizes including patterns that extend across the full length of the target. We have used two approaches for generating templates. A filter approach and a wrapper approach. In the filter approach, templates are synthesized and evaluated without examining their value relative to other features. To filter templates, performance measures are defined that evaluate individual features. We have developed several measures that determine the effect of a feature on the full set of targets. These measures typically seek features that fit consistently into several classes of targets while not fitting into samples of the remaining classes of targets. The problem is complicated

because targets can rotate and a template may fit into several classes of targets under some range of rotation and fit into other classes for a different range of rotations. With a wrapper approach, templates are synthesized to complement existing sets of templates. Thus, a set of template is gradually extended by searching for additional templates that respond to specific targets that are not already handled by templates in the set. In this work, we have adopted the filter method since it is faster to generate templates using filters, and it is a good match with our new feature selection algorithm.

4.3.5.2 Feature Selection and Classification

We have developed a new evolutionary learning algorithm that performs feature selection and classification. The morphological feature extraction process rapidly generates a large pool of candidate features. The problem is to select a subset of these features capable of robust target classification. Since each feature represents the presence or absence of a specific pattern in the voxelized targets the output of the feature is a simple binary value (0,1). A feature vector is simply a binary vector indicating the presence of absence of a subset of measures in the pool. Our task is to find a cooperative subset of features among the pool of available features. We use the genetic algorithm to search the space of potential features to optimize the choice of features. For this problem, we decide to use a nearest neighbor classifier (KNN with K=1) to assign a class label to each data sample. KNN is one of the most commonly used classifier and often produces a near optimal classification. There are two problems associated with the use of KNN classifiers. First, if the features have different ranges of values, the features must be weighted otherwise features with large values will overshadow features with small values. Since all of our features are binary, this is not a problem we have to worry about. The other problem with KNN is the approach depends on defining a set of prototypes feature vectors to represent each target class. Given a fixed set of features, there are several techniques for identifying a good set of prototypes. Since our set varies during the evolutionary search, we also use a genetic algorithm to search for a good set of prototypes. The basic idea is to create a population composed of members that contain a bit vector that selects a subset of features from the M available features and a second bit vector that selects a subset of prototypes (selected from the training samples) to form a population of KNN classifier.

The representation of each member of the population would appear as follows:

Pool of features:	1	2	3	4	5	6	7	8	9	10	11	12	…	(M-2)	(M-1)	M
Chromosome 1:	1	1	0	0	0	1	1	0	0	0	0	1		0	0	1

Potential prototypes:	1	2	3	4	5	6	7	8	9	10	11	12	…	(N-2)	(N-1)	N
Chromosome 2:	0	1	1	0	1	0	0	1	0	0	1	1		1	0	0

In this example, the member of the population composed of chromosomes 1 and 2 selects features 1, 2, 6, 7, 12, …, M and training samples 2, 3, 5, 8, 11, 12, … (N-2) to define the prototypes for the KNN classifier. This KNN classifier can then be assembled and used to evaluate the remaining training samples (samples not selected using chromosome 2). The accuracy on these samples determines the fitness of the features and prototypes. Basic evolutionary processing involving recombination and mutation is then used to mix member of the population to facilitate the search for improved classification systems.

4.3.5.3 Experimental Results

Notice how the genetic search alters the best classification system which begins using 50 features and 23 prototypes and slowly adjusts the classifier to use 20 features and 26 prototypes (see Figures 52 and 53). This demonstrates how the system works to reduce complexity while maintaining accuracy. The current learning procedure favors accuracy over the number of features and prototype. This means the system maximizes accuracy and only uses the other measures for comparing solutions with the same accuracy. We are currently working to adjust the fitness function used to drive the evolutionary process to balance the accuracy, number of features, and number of prototypes. Additional experiments show similar results for other combinations of targets and larger data set.

4.3.6 Discussion of E3D Results

The approach described here indicates that mathematical morphology is a powerful method for extracting features from 3D data. Using randomly generating morphological

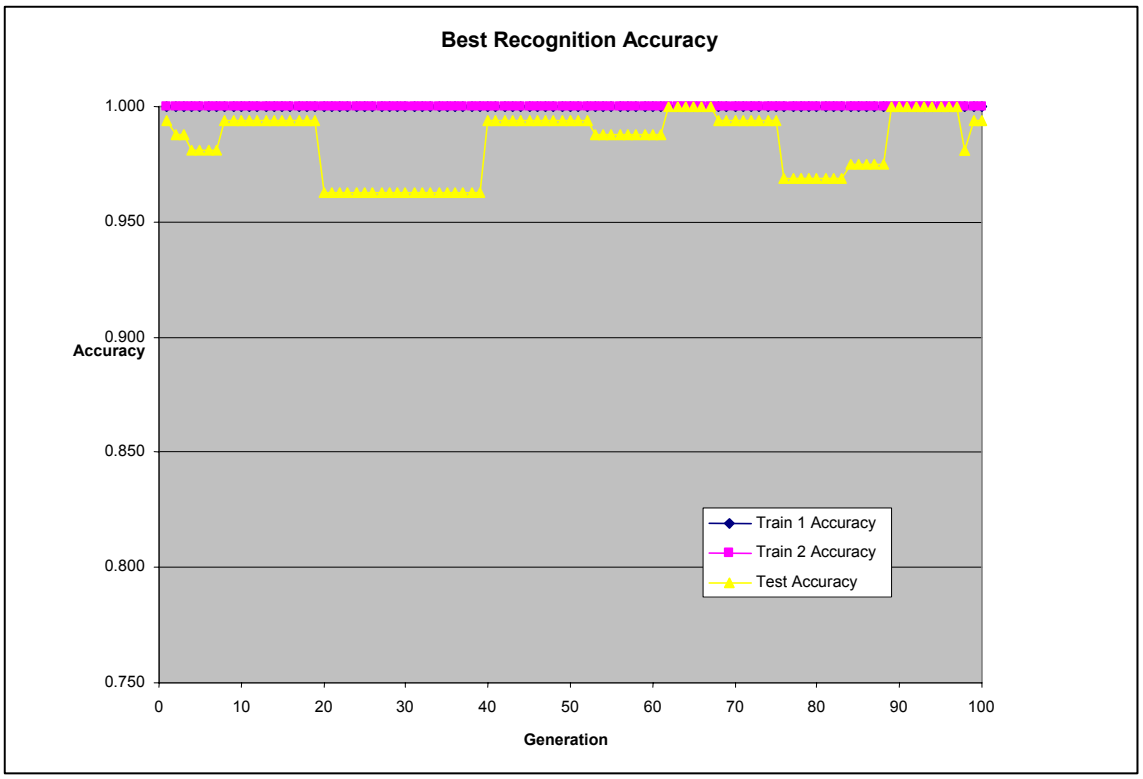


Figure 52. Best Recognition System's E3D Accuracy

sequences, reasonable ROC curves were obtained. These curves showed relatively slow degradation under changes to viewing angle and indistinguishable differences under varying noise levels and distance. We expect that these results would hold up under even more extreme noise and variation in distance between the target and sensor.

Several immediate avenues to be explored include the relationship between system performance, the scale used to voxelize the point cloud, and the length of the morphological sequences. Increasing the resolution of voxelization (e.g., from 0.4m to 0.2m) would increase the detail of the vehicles but would also make the system more sensitive to noise. Another negative aspect to increasing resolution is that the number of pixels in the images increases dramatically. For example, doubling the resolution will increase the size of the images by a factor of 8. Finding the appropriate trade-offs will be important. Also, allowing sequences to have more than 2 operations will increase the power and flexibility of the morphological processing; however, the number of such sequences increases exponentially, making the search process very difficult.

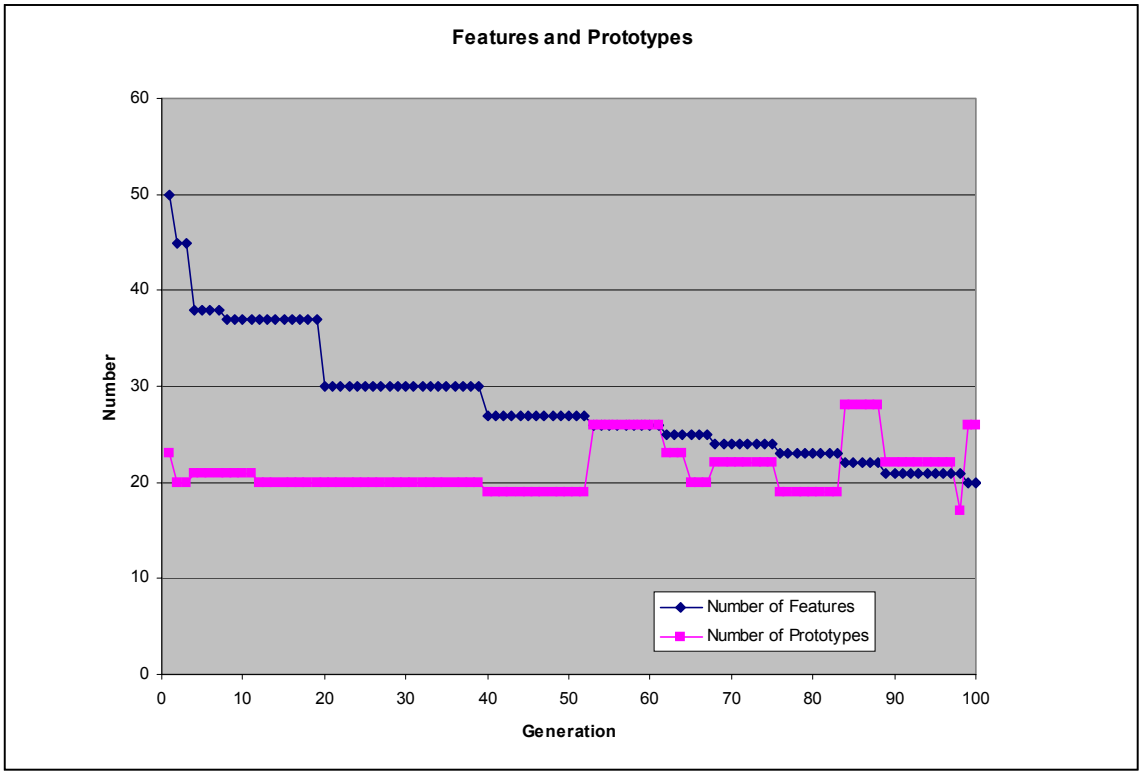


Figure 53. Best Recognition System's Complexity

4.4 EOC Application

The final phase of this research explored the effects of extended operating conditions (EOCs) on the performance of the HELPR architecture using images generated by Wright State personnel. By definition, extended operating conditions represents all possible types of variations to the physical structure and characteristics of a target, the myriad of environments where a target can appear, and the distortion, errors and loss of information due to the behavior of the sensor system. It is impossible to study the complete range of operating condition that can occur in an uncontrolled environment. The HELPR architecture is capable of combining different types of features (statistical, geometric, etc.) to form a recognition system so we chose to focus our attention on one type of feature -- color. Color is a very powerful feature that is becoming readily

accessible via digital cameras and video. It can be used for rapid screening of objects without the need to perform complex structural analysis of a large scene. It is a particularly a valuable aid for work in security where the color of objects such as cars or a suspects clothing can facilitate tracking and identification. The problem with color as a feature is that color is effected by the source of illumination and surface reflection. For example, consider the blue car shown in Figure 54. Here

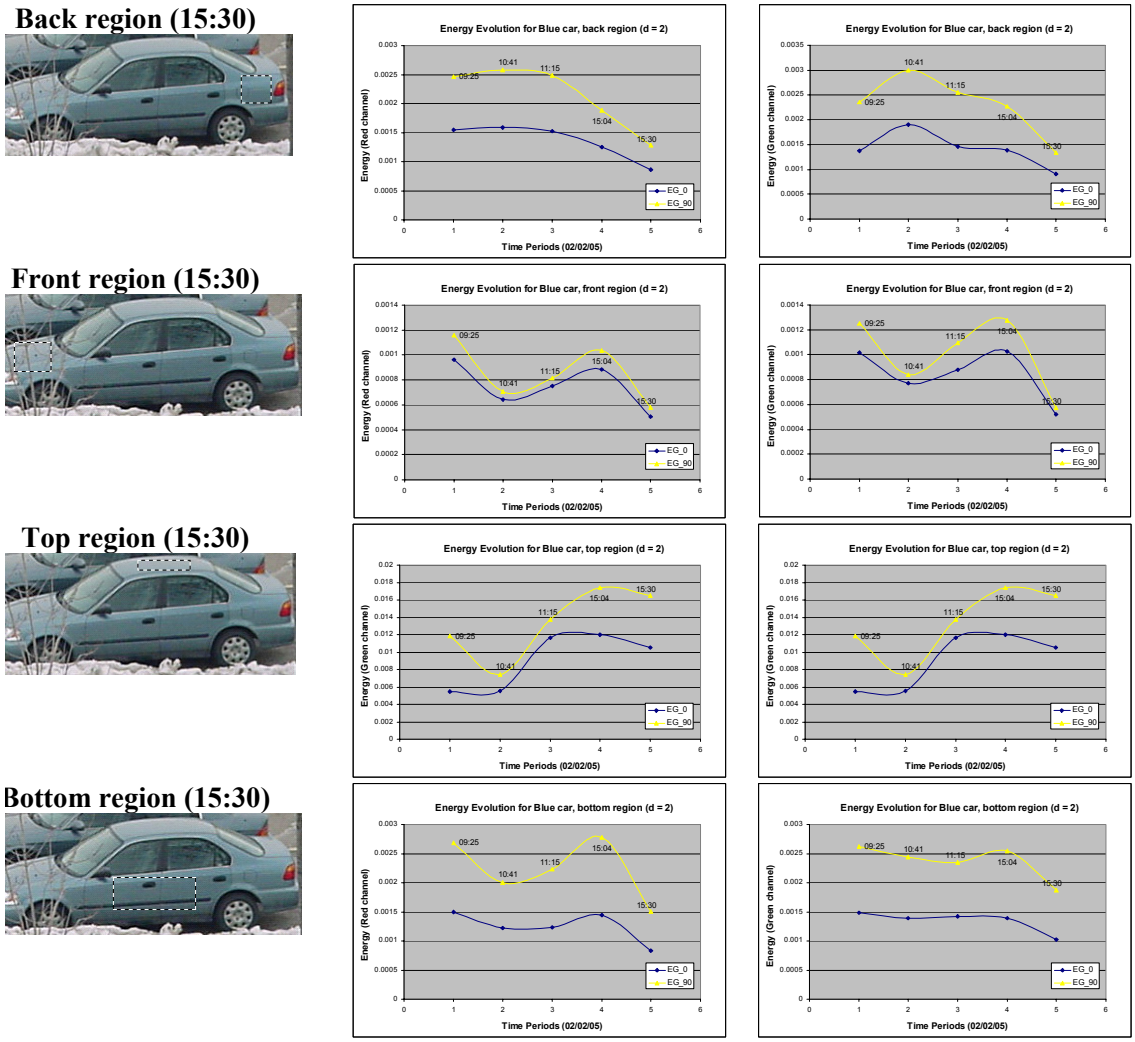


Figure 54. Sample Color Variation By Time of the Day

we see the variation in the RGB components of color as a function of the time of day. As the sun

moves through the sky and clouds pass by, the intensity of the color varies. If two cars with similar colors appear, we will not be able to distinguish one car from the other based on color.

There are two issues. The color of the source of the illumination can impact our perception of color. For example a red light shining on a white car produces the same color as a white light shining on a red car. In addition, reflection alters the perception of color. Again a white car parked next to a blue car looks different than a white car parked next to another white car.

The color constancy problem involve removing the effects of illumination by finding a color transformation that can be used to transform an image to a standard color reference. In essence this allows us to create EOC sensitive feature that adapts its behavior based on the EOC. To apply these feature, first measurements of the full scene are used to "parameterize" the recognition features that are used in the pattern recognition system.

4.4.1 Parking Lot Color Constancy Problem

The main objective of this investigation is to use evolutionary search techniques to exploit the neighborhood data to compensate for environmental variations that affect the use of color in automatic target recognition. We utilize a set of color images of cars in a parking lot taken at different times that insure differences in the environmental conditions. A number of target cars are tracked and sampled in multiple images. The pixel colors recorded by the digital camera depend on the characteristics of the incident illumination and the surfaces reflecting the incident light. Perceptually, we have unknown mental means for compensating for variation in illumination so object colors appear to be constant. When using colors as features for target recognition, there is need to compensate for variations in color measurements. Needed are computer algorithms to transform color measurements into a standard reference frame. Compensating for these changes is called the color constancy problem in the literature. Although a number of approaches to the color constancy problem have been published, this endeavourer is still an active research topic. See review articles in [6, 7].

To appreciate the inherent difficulties in using colors as features, consider a simple equa-

tion that models the incident and reflected light signals:

$$C^c = [R, G, B] = f(E^c(\lambda) \cdot S^c(\lambda) \cdot F^c(\lambda)) \qquad \text{Eq. 4.4-1}$$

where

C^c - three color channel measurements, c=(R,G,B).

$E^c(\lambda)$ - spectral power distribution of the illuminant,

$S^c(\lambda)$ - spectral reflectance function of the surface,

$F^c(\lambda)$ - sensor sensitivity functions for three color channels

Not shown are dependencies on geometric parameters and different reflectance processes.

Simplifying equation 1 still further, let $F^c(\lambda) = \delta(\lambda - \lambda^c)$ (delta functions) so that

$$C^c = E^c \cdot S^c \qquad \text{Eq. 4.4-2}$$

where E^c are the unknown scene illuminants and S^c the surface reflectance parameters.

In order to compare the camera signal with a color reference depending on the invariance of S^c, it is necessary to account for the variations in E^c. Obviously, the separation of C^c into two components E^c and S^c is an ill defined problem. For example, a red pixel measurement could result from white illumination reflecting off a red surface or red illumination reflecting off a white surface. The various approaches to and explanations for color constancy depend on using additional constraints and regularities between illuminants and surface characteristics. In this paper, we investigate methods for predicting unknown illuminants based on the distribution of color over the entire scene.

A simple adaptation model, which was proposed by Johannes von Kries in 1902 [65] to explain biological adaptation, has been used for most color constancy algorithms. This model, which forms the basis of many current approaches, states that the adaptation of the visual system to different illuminants is done by adjusting the gain coefficients associated with each of the color channels. It can be expressed in terms of a diagonal matrix that transforms the measurements under one illumination into the values that would be measured in another reference illumination,

$$\begin{vmatrix} R' \\ G' \\ B' \end{vmatrix} = \begin{vmatrix} T^R & 0 & 0 \\ 0 & T^G & 0 \\ 0 & 0 & T^B \end{vmatrix} \bullet \begin{vmatrix} R \\ G \\ B \end{vmatrix} \qquad \text{Eq. 4.4-3}$$

Utilizing $C^c = E^c \cdot S^c$ (equation 2), we observe that the diagonal elements are derivable from the illuminants E^c

$$T^R = \frac{E^{R'}}{E^R}, \; T^G = \frac{E^{G'}}{E^G}, \; T^B = \frac{E^{B'}}{E^B} \qquad \text{Eq. 4.4-4}$$

In our investigation, we use a two dimensional chromaticity space to reduce the number of parameters and to gain invariance with respect to illumination intensity. We use the following two definitions of chromaticity space:

$$\tilde{r} = \frac{R}{B}, \; \tilde{g} = \frac{G}{B} \qquad \text{Eq. 4.4-5a}$$

$$r = \frac{R}{(R+G+B)}, \; g = \frac{G}{(R+G+B)}, \; b = \frac{B}{(R+G+B)} \qquad \text{Eq. 4.4-5b}$$

where

$$r + g + b = 1 \qquad \text{Eq. 4.4-5c}$$

The two chromaticities are related by the following transformations

$$\tilde{r} = \frac{r}{b}, \; \tilde{g} = \frac{g}{b} \qquad \text{Eq. 4.4-6a}$$

$$r = \frac{\tilde{r}}{\tilde{r} + \tilde{g} + 1}, \; g = \frac{\tilde{g}}{\tilde{r} + \tilde{g} + 1}, \; b = \frac{1}{\tilde{r} + \tilde{g} + 1} \qquad \text{Eq. 4.4-6b}$$

Although both chromaticities are independent of the illumination intensity, the $\tilde{r}\tilde{g}$-chromaticity also transforms according to a diagonal matrix

$$\begin{vmatrix} \tilde{r}' \\ \tilde{g}' \end{vmatrix} = \begin{vmatrix} t_r & 0 \\ 0 & t_g \end{vmatrix} \bullet \begin{vmatrix} \tilde{r} \\ \tilde{g} \end{vmatrix} \qquad \text{Eq. 4.4-7}$$

where

$$t_r = \frac{T^R}{T^B} = \frac{R'}{B'} \cdot \frac{B}{R} = \frac{E^{R\,\prime}}{E^{B\,\prime}} \cdot \frac{E^B}{E^R} = \frac{e^{\tilde{r}'}}{e^{\tilde{r}}}, \quad t_g = \frac{T^G}{T^B} = \frac{G'}{B'} \cdot \frac{B}{G} = \frac{E^{G\,\prime}}{E^{B\,\prime}} \cdot \frac{E^B}{E^G} = \frac{e^{\tilde{g}'}}{e^{\tilde{g}}} \qquad \text{Eq. 4.4-8}$$

$$e^{\tilde{r}} \equiv \frac{E^R}{E^B}, \quad e^{\tilde{g}} \equiv \frac{E^G}{E^B} \qquad \text{Eq. 4.4-9}$$

Note that $C^c = E^c \cdot S^c$ and that the diagonal matrix elements are independent of the surface reflectance parameters S^c.

An excellent review of algorithms for estimating the illumination chromaticities *rg* are described by Barnard *et al.* [6, 7]. Forsyth's gamut-mapping method [18, 19] and Cardei *et al.*'s neural net method [11, 24, 25] are two prominent approaches that utilize the distribution of scene chromaticities. The distributions of chromaticities are derived by dividing the normalized chromaticities space (6.b) into a regular lattice of bins. A bin is set equal to one if at least one pixel chromaticity falls within its boundary and zero otherwise. We will refer to this distribution as a binary histogram. Figure 1.a shows a typical parking lot scene, 1.b the distribution of all pixel chromaticities, and 1.c the corresponding binary histogram for a 50 x 50 lattice. The next section discusses our use of different HELPR techniques for utilizing neural networks for predicting illuminant chromaticity from the distributions of image chromaticities.

4.4.1.1 Estimating Illuminant Chromaticity For Training Images

In Cardei [11] the grey world algorithm provides a relatively good estimation of the illuminant for images with lots of colors. These estimations are used in a novel training algorithm called "neural network bootstrapping" because the neural network in turn demonstrates better performance than the grey world algorithm used to train it.

Cardei used back-propagation to train a multilayer perceptron [11] with two hidden layers. The input to the network was effectively the binary histogram derived from all image chromaticities as described above and the network output the two illuminant chromaticity parameters r and g. The advantage of using a neural network is that there is no explicit assumption regarding scene content. The advantage of using the binary histogram as input is invariance to scene geometry such as surface sizes and orientations. The Von Kries color algorithm utilizes the output to compute the components for the diagonal color correction matrix.

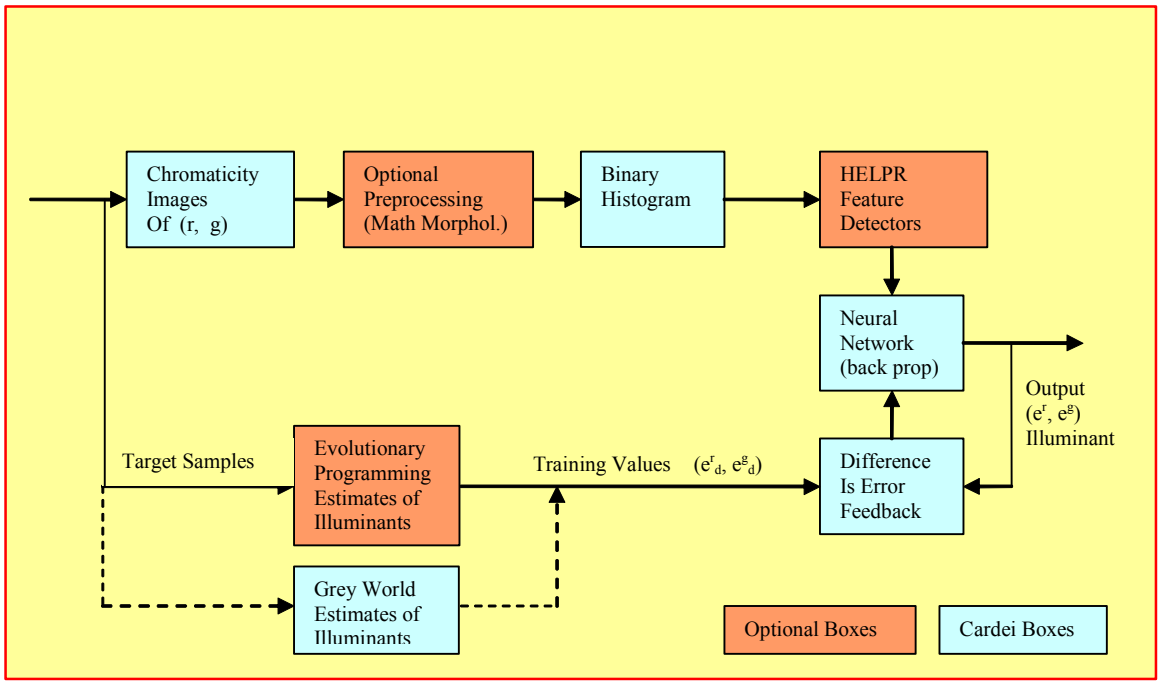


Figure 55. Predicting Illuminant Chromaticity.

This chart illustrates the HELPR extension of the neural network approach introduced by Cardei et el. HELPR uses evolutionary programming to generate the training set instead of using the gray world algorithm. HELPR techniques are applicable to preprocessing chromaticity images and synthesizing new feature vectors to replace the binary histograms as direct inputs to the neural network.

In this paper, we explore the application of HELPR techniques to generate a training set to enhance Cardei's neural network approach for predicting image illuminants. As illustrated in figure 55, we present new methods for generating estimates of the training illuminants and for generating feature vectors for inputs to the neural network instead of a vectorized binary histogram.

We explored an alternative to using the grey world algorithm for estimating illuminants $\left(e^{\tilde{r}},\ e^{\tilde{g}}\right)$ from the real images used in the experiment. We also made Cardei's simplifying assumption that the illuminant chromaticity is constant for each parking lot image. A number of vehicle targets were manually identified and sampled across multiple images taken at different times to insure different environmental conditions. The two phase procedure was used consisting of the sampling phase and an evolutionary search for the illuminant chromaticities for each sampled image. In each image, multiple RGB samples were taken on the roof and hoods of each target the image. Not all targets appeared in all images. This data collection was used with an evolutionary search to generate an optimal estimate of the image illuminants.

An evolutionary programming algorithm (EPA) is utilized for the evolutionary search [21]. An EPA utilizes a population of chromosomes which undergo mutations and selection over many cycles. The chromosomes are vectors of search parameters which are the illuminants $\left(e^{\tilde{r}},\ e^{\tilde{g}}\right)$ for the sampled images. The EPA is a relatively simple algorithm requiring an appropriate performance measure PM to guide the evolutionary search. We present three different performance measures called *all-means*, *all-pairs*, and *relative means*. Given the image illuminants defined by a chromosome, a PM represents how effective the correction algorithm relates the sample chromaticity in different images.

The vehicle chromaticity samples in each image have a distribution about their centroid. The color Von Kries correction algorithm can only distort these distributions when transforming them to a different reference frame. The ideal set of chromosomes is one that will transform all vehicle centroids into a common reference frame so that they map into the same point. Basically this is the optimal solution that achieves the tightest cluster and provides the most effective cor-

rection for a pattern recognition color feature.

One trivial solution achieves a very tight cluster by mapping all samples into the origin which is avoided in our definitions by the use of chromaticity references to anchor the search process. The three different PMs use different methods for commuting reference parameters and defining average cluster sizes over all vehicles based on the Euclidian distances between samples and the references.

One needs to take care in the use of normalized and un-normalized chromaticities. The general rule is that when using the Von Kries correction algorithm, one uses un-normalized chromaticities. For the neural network I/O it is convenient to use normalized chromaticities with values between 0 and 1. Equations 7a and 7b relate the two formats.

In the following PM definitions, let the red and green target sample chromaticities be denoted by (*r(tjs)*, *g(tjs)*) where *t* , *j*, and *s* are indices for the target, image, and s^{th} sample in the *j*th image. In addition let sample clusters be specified by:

$t = 1, 2, .., L,$	L = number of targets,
$j = 1, 2,\ldots, Mt,$	Mt = number of images associated with target t,
$s = 1, 2,.., Ntj,$	Ntj = cluster size for target t in j^{th} image.

Let the illumination chromaticities for the j^{th} images be denoted by (e^r_j, e^g_j). The set over all images make up the EP chromosome. The diagonal correction elements (t^r, t^g), which are defined in terms of (e^r_i, e^g_i), will be denoted by:

$$Kij \equiv t^r{}_{ij} = \frac{e^r{}_i}{e^r{}_j}, \quad \tilde{K}ij \equiv t^g{}_{ij} = \frac{e^g{}_i}{e^g{}_j}, \; i \neq j$$

.

These elements are used to compare sample chromaticities between different images. For example, to transform the value of (*r(tjs)*, *g(tjs)*) from image *j* into the image *i* we use

the following notation:

$$r(tijs) \equiv \frac{e^{r}{}_{i}}{e^{r}{}_{j}} \cdot r(tjs) \equiv Kij \cdot r(tjs),\ g(tijs) \equiv \frac{e^{g}{}_{i}}{e^{g}{}_{j}} \cdot g(tjs) \equiv \tilde{K}ij \cdot g(tjs),\ i \neq j$$

.

4.4.1.2 All Means-PM Definition

The following notation is used to define residual errors for image-to-image transformations:

$$dr(tijs) \equiv r(tijs) - \hat{r}(ti) = Kij \cdot r(tjs) - \hat{r}(ti),\ i \neq j,$$

where we define a reference chromaticity (red) for target t in i^{th} image by

$$\hat{r}(ti) = \frac{\sum_{s=1}^{Nti} r(tis)}{Nti} \qquad \text{(similar expression for green component)}$$

Define the cluster error relative to $\hat{r}(ti)$ by :

$$dr(tij) = \sum_{s=1}^{Ntj} \{Kij \cdot r(tjs) - \hat{r}(ti)\}^2 = Kij^2 \cdot \sum_{s=1}^{Ntj} r(tjs)^2 - 2 \cdot Kij \cdot \hat{r}(ti) \cdot \sum_{s=1}^{Ntj} r(tjs) + \hat{r}(ti)^2 \cdot Ntj,\ i \neq j$$

Rewriting

$$dr(tij) = Kij^2 \cdot Atj + Kij \cdot Btij + Ctij,\ i \neq j$$

where

$$Atj = \sum_{s=1}^{Ntj} r(tjs)^2,\quad Btij = -2 \cdot \hat{r}(ti) \cdot \sum_{s=1}^{Ntj} r(tjs),\quad Ctij = \hat{r}(ti)^2 \cdot Ntj.$$

Also, $dg(tij) = \tilde{K}ij^2 \cdot \tilde{A}tj + \tilde{K}ij \cdot \tilde{B}tij + \tilde{C}tij$, so the combined error which is the Euclidian residual about the means is defined by

$$dD^2tij = dr(tji) + dg(tji),$$

$$dD^2tij = Kij^2 \cdot Atij + \tilde{K}ij^2 \cdot \tilde{A}tj + Kij \cdot Btij + \tilde{K}ij \cdot \tilde{B}tij + Ctij + \tilde{C}tij.$$

The remaining PM definitions have residual error terms with similar formats using pre-computed

ABC matrices which are functions of only the sampled target chromaticities.

Using the all cluster means as a reference, the '*all means*' performance measure PM is the average over all the cluster means defined by

$$PM = \frac{\sum_{t=1}^{P}\sum_{i=1}^{Mt}\sum_{j=1,j\neq i}^{Mt} dD^2 tij}{Nterms} = \frac{\sum_{t=1}^{P}\sum_{i=1}^{Mt}\sum_{j=1,j\neq i}^{Mt}(dr(tij)+dg(tij))}{Nterms} \quad (j \neq i) \ ,$$

where $Nterms = \sum_{t=1}^{Ntargets}\sum_{i=1}^{Mt}\sum_{j=i,j\neq i}^{Mt} Ntj$.

4.4.1.3 All Pairs-PM Definition

For each target, the set of all pairs of measurements associated with a target are considered. The set of image illuminations (e^r_j, e^g_j) are used to transform one member of the pair into the other member's image in order to compute the pair's residual error in the same image. The notation for this definition begins with:

$$\Delta r(tij) = \sum_{\mu=1}^{Nti}\sum_{s=1}^{Ntj}(Kij \cdot r(tjs) - r(ti\mu))^2, \ i \neq j$$

$$\Delta r(tij) = \{Kij^2 \cdot \langle r^2(tj)\rangle - 2 \cdot Kij \cdot \langle r(ti)\rangle \cdot \langle r(tj)\rangle + \langle r^2(ti)\rangle\}(Nti \cdot Ntj), \ i \neq j$$

where $\langle r^2(tj)\rangle = \frac{\sum_{s=1}^{Ntj} r(tjs)\cdot r(tjs)}{Ntj}$ *and* $\langle r(tj)\rangle = \frac{\sum_{s=1}^{Ntj} r(tjs)}{Ntj}$.

We may rewrite

$$\Delta r(tij) = Kij^2 \cdot Atj + Kij \cdot Btij + Ctij, \ i \neq j \ ,$$

where

$$Atij = \left\langle r^2(ti) \right\rangle \cdot Nti \cdot Ntj, \quad Btij = -2 \cdot \left\langle r(i\mu) \right\rangle \cdot \left\langle r(is) \right\rangle \cdot Nti \cdot Ntj, \quad Ctij = \left\langle r^2(tj) \right\rangle \cdot Nti \cdot Ntj .$$

Let

$$PM = \frac{\sum_{t=1}^{P} \sum_{i=1}^{Mt} \sum_{j=1, j\neq i}^{Mt} (\Delta r(tij) + \Delta g(tij))}{Nterms} \quad (j \neq i),$$

where

$$Nterms = \sum_{t=1}^{P} \sum_{i=1}^{Mt} \sum_{j\neq i}^{Mt} Nti \cdot Ntj .$$

4.4.1.4 Relative Means – PM Definition

The next definition is similar to the *all pairs* PM except that it uses only the target centroids to represent all target samples.

$$\delta r(tij) = \{mean(Kij \cdot r(tjs) - \hat{r}(ti))\}^2 = \{Kij \cdot \hat{r}(tj) - \hat{r}(ti)\}^2$$

$$PM = \frac{\sum_{t=1}^{P} \sum_{i=1}^{Mt} \sum_{j>i}^{Mt} \delta D^2 tij}{Nterms} = \frac{\sum_{t=1}^{P} \sum_{i=1}^{Mt} \sum_{j>i}^{Mt} (\delta r(tij) + \delta g(ti)j)}{Nterms} \quad (j \neq i),$$

where $Nterms = \sum_{t=1}^{P} \frac{Mt \cdot (Mt-1)}{2}$.

4.4.1.5 Predicting Illuminant Chromaticity Using Neural Networks

After the optimal illuminants are found with the use of EP, they are used to train a multi-layer perceptron to transform the binary histogram of the image illuminants into the image illumi-

nants. The bin size used to define the binary histogram was a major consideration in Cardei's thesis [11]. A high resolution histogram with many bins increases the computational load for training the neural network, whereas too few bins introduce prediction errors. The bin size was determined by repeatedly training the neural network. The HELPR approach uses a high resolution binary histogram to synthesize a feature vector for input to the neural network instead of the vectorized binary histogram. Any holes resulting from using small bin sizes may be filled in with a mathematical morphological operator called a 'closing operator'. Thus, HELPR improves the training algorithm by synthesizing a smaller customized feature vector using the highest resolution histogram. This aspect of the investigation is characteristic of the use of HELPR in automatically generating pattern recognition systems.

5.0 Summary and Discussion

The goal of the HELPR project was to create techniques and tools for evolving complete pattern recognition systems. Although there are systems available for forming pattern recognition system when you are given a set of working features, there are still very few systems available that extract, synthesize and select features to form complete pattern recognition system. The results from our experiments have demonstrated that the HELPR architecture is sufficiently flexible to adapt to different types of ATR related recognition problems. One of the criticisms of "weak" machine learning techniques is that although they can solve a wide variety of problems, they often produce results that are not quite as good as those produced by human experts. This old way of thinking is slowly beginning to give way to a new approach called memetic algorithms that combines machine learning systems with human expertise to create new tools that have the advantage that they are applicable to a wide range of problem and can produce solutions as good or better than human experts. We believe that the HELPR architecture could form the foundation for a memetic system capable of solving ATR problems faster and more accurately than possible using pure human expertise.

6.0 References

1. Ackley, D. H. "Stochastic Iterated Genetic Hillclimbing," Ph.D. Dissertation, Carnegie Mellon University, 1987.
2. Air Force Research Laboratory Sensors Directorate draft technical description of challenge problem, April, 2003.
3. Anderson, J. and E. Rosenfeld. Neurocomputing: Foundations of Research, Massachusetts Institute of Technology, 1988.
4. Arts, E. and J. Korst. Simulated Annealing and Boltzman Machines. Poland Wiley, 1989.
5. Back, T., Evolutionary Algorithms in Theory and Practice, Oxford University Press, 1996.
6. Barnard, Korbus, Vlad Cardei, and Brian Funt, A Comparison of Computational Color Constancy Algorithms—Part I: Methodology and Experiments With Synthesized Data, IEE Trans. Image Processing, Vol. 11, No. 9, September 2002.
7. Barnard, Kobus, Lindsay Martin, Adam Coath, and Brian Funt, A Comparison of Computational Color Constancy Algorithms—Part II: Experiments With Image Data, IEE Trans. Image Processing, Vol. 11, No. 9, September 2002
8. Battiti, R., “Using Mutual INformation for Selecting Features in Supervised Neural Network Learning,” IEEE Trans. Neural Networks, Vol. 5, pp. 537-550, 1994.
9. Bischel, M. and P. Seitz, “Minimum Class Entropy: Maximum Information Approach to layered Networks,” Neural Networks, Vol. 2, pp. 131-141, 1989.
10. Burden, R. L. and J. D. Faires. Numerical Analysis. Prindle, Weber, Schmidt, 1985.
11. Cardei, V.. (2000) *A neural network approach to color constancy.* PhD_Thesis, Simon Fraser Univ., Burnaby, BC, Canada.
12. Courte, D., M. M. Rizki, L. A. Tamburino, "Evolving gaussian windowing functions for data preprocessing in an odor classification system". 12th Midwest Conference on Artificial Intelligence and Computer Science, pp 58-65, 2001.
13. de Hass, L. J. "Automatic Programming of Machine Vision Systems," 10th International Joint Conference on Artificial Intelligence, pp. 790-792.
14. Dougherty, E., An Introduction to Morphological Image Processing. Bellingham, WA: SPIE, 1992.
15. Duda, R. O. and P. E. Hart. Pattern Classification and Scene Analysis, John Wiley and Sons, 1973.
16. Duda, R., P. Hart, and D. Stork, Pattern Classification, New York: Wiley, 2001.
17. Fahlman, S. E. and C. Lebiere. "The Cascade-Correlation Learning Architecture," CMU-CS-90-100, 1990.
18. Finlayson, G. D. , “Coefficient color constancy,” Ph.D. dissertation, Sch. Comput., Simon Fraser Univ., Burnaby, BC, Canada, 1995.

19. Finlayson, G. D. , “Color in perspective,” *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 18, pp. 1034–1038, 1996.

20. Fisher, R. A. "The Use Of Multiple Measures In Taxonomic Problems," Ann Eugenics, 7, Part I, pp. 179-188, 1936.

21. Fogel, D. B. System Identification Through Simulated Evolution. Ginn Press, 1991.

22. Fogel, L. J., A. J. Owens, and M. J. Walsh. "On the Evolution of Artificial Intelligence," Proceedings of the 5th National Symposium on Human Factors in Electronics, pp. 63-76.

23. Fukushima, K., “Neocognitron: A Self-organizing Neural network Model for Pattern Recognition Unaffected by Shift and Position,” Bio. Cybernetics, Vol. 36, pp. 193-202, 1981.

24. Funt, B., V. Cardei, and K. Barnard, *“Learning color constancy,”* in *Proc. IS&T/SID 4th Color Imaging Conf.: Color Science, Systems, and Applications*, Scottsdale, AZ, 1996, pp. 58–60.

25. Funt, B. V., V. C. Cardei, and K. Barnard, *“Method of estimating chromaticity of illumination using neural networks,”* U.S. Patent 5 907 629, 1999.

26. Gabor, D. "Theory of Communication," J.I.E.E., pp. 429-459, 1946.

27. Gillies, A. M.. "Machine Learning Procedures for Generating Image Domain Feature Detectors," Ph.D. Dissertation, University of Michigan, 1985.

28. Goldberg, D. E. Genetic Algorithms in Search, Optimization, and Machine Learning. Addison Wesley, 1989.

29. Holland, J. H. Adaptation in Natural and Artificial Systems. The University of Michigan Press, 1975.

30. Haralick, R. M. S. R. Sternburg, and X. Zhuang, “Image Analysis Using Mathematical Morphology,” IEEE Trans. on Pattern Analysis and Machine Intelligence, PAMI-9, pp. 532-550, 1987.

31. Haykin, Simon, *Neural Networks, A Comprehensive Foundation*, Prentice Hall, 1999.

32. Hwang, S. Y. "State-Space Search for High-Level Control of Machine Vision," Optical Engineering, Vol. 31, No. 6, pp. 1264-1276, 1992.

33. Joo, H., R. M. Haralick, and L. G. Shapiro. "Toward the Automatic Generation of Mathematical Morphology Procedures Using Predicate Logic," 3rd International Conference on Computer Vision, pp. 156-165, 1990.

34. Katz, A. J. and P. R. Thrift, "Generating Image Filters for Target Recognition by Genetic Learning," IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 16, No. 9, pp. 906-910, 1994.

35. Koskinen, L., J. Astola, and Y. Neuvo, "Soft Morphological Filters," SPIE Image Algebra and Morphological Image Processing II, pp. 262-270, 1991.

36. Koza. J. Genetic Programming: On The Programming of Computers by Means of Natural Selection. MIT Press, 1992

37. Matsuyma, T. "Expert Systems for Image Processing: Knowledge-Based Composition of Image Analysis Processes," Computer Vision, Graphics, and Image Processing, Vol. 48, pp. 22-49, 1989.

38. Minsky, M. L. and S. A. Papert. Perceptrons. MIT Press, 1988.

39. Pao, Y. H. Adaptive Pattern Recognition and Neural Networks. Addison-Wesley, 1989.

40. Rizki, M., L. Tamburino, and M. Zmuda, "Adaptive Search for Morphological Feature Detectors," SPIE Image Algebra and Morphological Image Processing I, pp. 150-159, 1990.

41. Rizki, M. M., Tamburino, L. A., and Zmuda, M. A., Applications of learning strategies to pattern recognition, SPIE Applications of Artificial Intelligence and Neural Networks, pp. 384-391, 1991a.

42. Rizki, M. M., Tamburino, L. A., and Zmuda, M. A., Biological evolution as a paradigm for performance driven design processes in Lecture Notes in Computer Science: Computing in the 90's , Springer Verlag, New York, pp 85-91, 1991b.

43. Rizki, M. M., Tamburino, L. A., and Zmuda, M. A., Multi-resolution correlators for target recognition, IEEE Canadian Conference on Electrical and Computer Engineering, pp. TA3.18.1-TA3.18.4, 1992.

44. Rizki, M. M., Tamburino, L. A., and Zmuda, M. A., Evolving multi-resolution feature detectors, IEEE NNC Second Annual Conference on Evolutionary Programming, pp. 108-118, 1993.

45. Rizki, M. M., Tamburino, L. A., and Zmuda, M. A., E MORPH: A Two-phased Learning System for Synthesizing Complex Morphological Classification Systems. IEEE NNC Third Annual Conference on Evolutionary Programming, pp 60-67, 1994.

46. Rizki, M. M., Tamburino, L. A., and Zmuda, M.A, Evolving Morphological Classification Systems, IEEE NNC Fourth Annual Conference on Evolutionary Programming, MIT Press, pp 95-106, 1995.

47. Rizki, M. and L. Tamburino, "Evolutionary Computation Applied to Pattern Recognition," Proceedings of the Third Annual Genetic Programming Conference, pp. 777-785, 1998.

48. Rizki, M., M. Zmuda, and L. Tamburino. “Evolving Pattern Recognition Systems,” IEEE Transactions on Evolutionary Computation, pp. 594-609, Dec. 2002.

49. Schmitt, M. "Mathematical Morphology and Artificial Intelligence: An Automated Programming System," Signal Processing, Vol. 16, pp. 389-401, 1989.

50. Schwefel. H. P. Numerical Optimization of Computer Models. John Wiley and Sons, 1981.

51. Serra, J., Image Analysis and Mathematical Morphology, Academic Press, 1982.

52. Stentiford, F.W.M. "Automatic Feature Design for Optical Character Recognition Using an Evolutionary Search Procedure," IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol 7, No 3, pp. 349-355, 1985.

53. Tamburino, L. A., Rizki, M. M., and Zmuda, M. A., Computational resource management in supervised learning systems. IEEE Proceedings of the National Aeronautics and Electronics Conference 89, pp. 1074-1079, 1989a.

54. Tamburino, L. A., Rizki, M. M., and VanValkenburgh, W., Automated feature detection using evolutionary learning processes, IEEE Proceedings of the National Aeronautics and Electronics Conference 89, pp. 1080-1087, 1989b.

55. Tamburino, L. A. and M. M. Rizki, Automatic generation of binary feature detectors, IEEE Aerospace and Electronic Systems Magazine, pp. 20-29, September, 1989c.

56. Tamburino, L. A. and Rizki, M. M., Applications of hybrid learning to automated system design, IEEE Proceedings: AI, Simulation and Planning in High Autonomy Systems, pp. 176-183, 1990.

57. Tamburino, L. A. and Rizki, M. M., Performance driven autonomous design of pattern recognition systems, International Journal of Applied Artificial Intelligence, 6, pp. 59-77, 1992a.

58. Tamburino, L. A. and Rizki, M. M., Feature extraction using morphological analysis of multiresolution grey scale images, SPIE Conference on Signal Processing, Sensor Fusion, and Target Recognition, pp. 385-397, 1992b.

59. Tamburino, L. A., Zmuda, M. A., and M. M. Rizki, Applying evolutionary search procedures to pattern recognition problems, IEEE NNC Second Annual Conference on Evolutionary Programming, pp. 183-191, 1993.

60. Tamburino, L., M. Rizki, and M. Zmuda, "Generating Pattern Recognition Systems Using Evolutionary Learning," IEEE Expert, pp. 63-68, Aug., 1995.

61. Tamburino, L. A., and Rizki, M. M., Resource Allocation for a Hybrid Evolutionary Learning System used for Pattern Recognition. IEEE NNC Fifth Annual Conference on Evolutionary Programming, MIT Press, pp 207-216, 1997.

62. Uhr, L. and C. Vossler, "A Pattern-Recogntion Program that Generates, Evaluates, and Adjusts its Own Operators," Computers and Thought, McGraw-Hill, 1963.

63. Verly, J. G. and R. L. Delanoy, "Adaptive Mathematical Morphology for Range Imagery," IEEE Transactions on Image Processing, Vol. 2, No. 2, pp. 272-275, 1993.

64. Vogt, R. C. "Automatic Generation of Morphological Set Recognition Algorithms," Ph.D. Dissertation, University of Michigan, 1988.

65. von Kries, J., “Chromatic Adaptation,” Festschrift der Albrecht-Ludwig-Universität, Firbourg, 1902. [in D.L. MacAdam, 1970]

66. Wicker, D., M. Rizki, and L. Tamburino, A Hybrid Evolutionary Learning System for Synthesizing Neural Network Classifiers, IEEE-NNC Seventh Annual Conference on Evolutionary Programming, 1997.

67. Wicker, D. "Evolutionary Synthesis of Feedforward Neural Networks," Ph.D. Dissertation, Wright State University, 1998.

68. Wilson, S. S. "Training Structuring Elements in Morphological Networks," Mathematical Morphology in Image Processing, edited by Edward Dougherty, Marcel Dekker, Inc., 1993.

69. Zadeh, L. "Fuzzy Sets," Information and Control, Vol. 8, pp. 338-353, 1965

70. Zmuda, M. A. "Automatic Generation of Morphological Image Processing Programs," Ph.D. Dissertation, Wright State University, 1992a.

71. Zmuda, M. A., Rizki, M. M., and Tamburino, L. A., Automatic generation of morphological sequences, SPIE Conference on Image Algebra and Morphological Image Processing III, pp. 106-118, 1992b.

72. Zmuda, M., L. Tamburino, and M. Rizki, "An Evolutionary Learning System for Synthesizing Complex Morphological Filters," IEEE Transactions on Systems, Man, and Cybernetics, pp. 645-653, Aug., 1996.

7.0 Appendix

7.1 Publication Produced Under HELPR

M. A. Zmuda, M. M. Rizki, and L. A. Tamburino, Automated Synthesis of Pattern Recognition Systems, Neural Networks, Fuzzy Logic, Evolutionary Programming, Data Mining, and Complex Systems, Vol. 9, ASME Press, New York, 1999, pp. 381-386.

D. Wicker, M. M. Rizki, and L. A. Tamburino, Multi-Tiered Tournament Selection for Evolutionary Synthesis, IEEE Conference on Evolutionary Computation and Neural Networks, 2000, pp. 207-215.

Tamburino, L. A., M. M. Rizki, and M.A. Zmuda, HELPR Evolved Pattern Recognition Systems, 5th World Multiconference on Systemics, Cybernetics and Informatics: Image, Acoustic, Speech and Signal Processing Part II, Vol. XIII, 2001, pp. 212-217.

Courte, D., M. M. Rizki, L. A. Tamburino, Evolving gaussian windowing functions for data preprocessing in an odor classification system. 12th Midwest Conference on Artificial Intelligence and Computer Science, 2001 pp 58-65.

Rizki, M. M., Zmuda, M. A., Tamburino, L. A., Evolving Pattern Recognition Systems, IEEE Transactions on Evolutionary Computation, Volume: 6 Issue: 6 , Dec 2002, pp: 594 -609

Zmuda, M. A., M. M. Rizki, Tamburino, L. A., Optimizing linear discriminants using evolutionary learning, ANNIE 2002: Proceeding of the Artificial Neural Network in Engineering Conference, ASME Press, 2002, pp. 261-266.

Courte, D., Rizki, M. M., Tamburino, L. A., Evolving Cooperative Recognition Systems, ANNIE 2002: Proceeding of the Artificial Neural Network in Engineering Conference, ASME Press, 2002, pp. 219-224.

Wicker, D., M.M. Rizki, Tamburino, L. A., E-Net evolutionary neural network systhesis, Journal of Neural Computing, Elsevier Vol 42:1-4 2002, pp 171-196.

Courte, Dale E., Mateen Rizki, Louis A. Tamburino, Ricardo Gutierrez-Osuna, Evolutionary Optimization of Gaussian Windowing Functions for Data Preprocessing. International Journal of Artificial Intelligence Tools, World Scientific Press, Volume: 12 Issue: 1, 2003, Pages 1-16.

Zmuda Michael A., Mateen M. Rizki, Louis A. Tamburino, Hybrid Evolutionary Learning for Synthesizing Multi-Class Pattern Recognition Systems, Applied Soft Computing, Volume 2, Issue 4, February 2003, Pages 269-282.

Courte, D. E., M. Rizki, L. A. Tamburino, Local Feedback in the Evolution of Complementary Feature Subsets, ANNIE 2003: Proceeding of the Artificial Neural Network in Engineering Conference, ASME Press, 2003, pp. 261-266.

Courte, D. E., M. Rizki, L. A. Tamburino, Evolutionary Optimization of Heterogeneous Classifiers, IC-AI'03: The 2003 International Conference on Artificial Intelligence, IC-AI 2003: 763-768.

Zmuda, M. A., M. M. Rizki, L. A. Tamburino, Mutating Real-Valued Vectors Using Angular Displacement, International Journal of Artificial Intelligence Tools, World Scientific Press, Volume: 12 Issue: 4, 2003, Pages 509-526.

Zmuda, M. A., M. Rizki, L. A. Tamburino, Target Classification Using Morphological Features, The 8Th World Multiconference on Systemics, Cybernetics, Informatics (SCI 2004)

Addtional Papers In Review